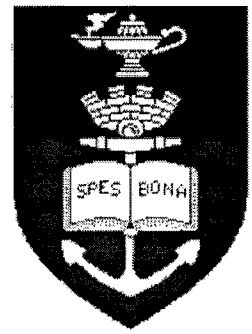


A Framework for the Analysis and Evaluation of Enterprise Models

Jean-Paul W. G. D. Van Belle

Thesis Submitted

for the Degree of Doctor of Philosophy

to the Department of Information Systems

University of Cape Town

February 2003

Copyright Waiver

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

.....
Jean-Paul Van Belle

Abstract

The purpose of this study is the *development and validation of a comprehensive framework for the analysis and evaluation of enterprise models*.

The study starts with an extensive literature review of modelling concepts and an overview of the various reference disciplines concerned with enterprise modelling. This overview is more extensive than usual in order to accommodate readers from different backgrounds.

The proposed framework is based on the distinction between the *syntactic*, *semantic* and *pragmatic* model aspects and populated with evaluation criteria drawn from an extensive literature survey.

In order to operationalize and empirically validate the framework, an exhaustive *survey of enterprise models* was conducted. From this survey, an XML database of more than twenty relatively large, publicly available enterprise models was constructed. A strong emphasis was placed on the interdisciplinary nature of this database and models were drawn from ontology research, linguistics, analysis patterns as well as the traditional fields of data modelling, data warehousing and enterprise systems. The resultant *database* forms the test bed for the detailed framework-based analysis and its public availability should constitute a useful contribution to the modelling research community.

The bulk of the research is dedicated to implementing and validating specific analysis techniques to quantify the various model evaluation criteria of the framework. The aim for each of the analysis techniques is that it can, where possible, be automated and generalised to other modelling domains.

The *syntactic* measures and analysis techniques originate largely from the disciplines of systems engineering, graph theory and computer science. Various metrics to measure model size, hierarchy, architecture and complexity are tested and discussed. It is found that many are not particularly useful or valid for enterprise models. Hence some new measures are proposed to assist with model visualization and an original "model signature" consisting of three key metrics is proposed.

Perhaps the most significant contribution of the research lies in the development and validation of a significant number of *semantic analysis techniques*, drawing heavily on current developments in lexicography, linguistics and ontology research. Some novel and interesting techniques are proposed to measure, *inter alia*, domain coverage, model genericity, quality of documentation, perspicuity and model similarity. Especially model similarity is explored in depth by means of various similarity and clustering algorithms as well as ways to visualize the similarity between models.

Finally, a number of *pragmatic* analyses techniques are applied to the models. These include face validity, degree of use, authority of model author, availability, cost, flexibility, adaptability, model currency, maturity and degree of support. This analysis relies mostly on the searching for and ranking of certain specific information details, often involving a degree of subjective interpretation, although more specific quantitative procedures are suggested for some of the criteria.

To aid future researchers, a separate chapter lists some promising analysis techniques that were investigated but found to be problematic from methodological perspective. More interestingly, this chapter also presents a very strong conceptual case on how the proposed framework and the analysis techniques associated with its various criteria can be applied to many other information systems research areas. The case is presented on the grounds of the underlying isomorphism between the various research areas and illustrated by suggesting the application of the framework to evaluate web sites, algorithms, software applications, programming languages, system development methodologies and user interfaces.

Acknowledgements

This work would not have been possible without the unwavering emotional, motivational and practical support from my dear wife Eva – words will never be able to express my thanks and appreciation. I also thank my lovely children Anneke, Jonathan and Sylvia for their patience and understanding, especially in the last phases of the thesis where daddy was all work and no play. Only people who have been through the "Ph D" experience themselves, can appreciate the sacrifices that close and loved family members go through in order for the dissertation to be written. Without my family's steadfast support, this research would have been still-born and this dissertation represents almost as much the culmination of their hard work as mine.

Of course, I must also thank my parents, Roger and Monica Van Belle-Deleu, who gave me my life, education and planted the seeds of all my inspirations.

Thanks also to my supervisors, without whom this thesis would have been stillborn: Prof Jonathan Miller who fired the starting gun, Prof Paul Licker who counted the laps, and Prof Mike Hart who manned the finish line. I must also express my gratitude to Prof Dewald Roode who made numerous valuable suggestions.

My colleagues from the information systems department here at UCT deserve a very special mention: Prof Derek Smith, Mike Eccles, Graham McLeod, and Jane Nash - for their support, guidance and practical assistance. Many deans who saw me through and provided financial supported whenever I called on them: from my years at UWC Prof André Kritzinger and Prof Renfrew Christie and here at UCT: Prof Peter Kritzinger, Prof Brian Kantor and Prof Douglas Pitt.

The following people shared their time, office, knowledge and often granted me hospitality at their homes: Tom Russ (Information Sciences Institute, University of Southern California), Henry Kim (Industrial Engineering, University of Toronto), Wouter Jansweijer (Social Science Informatics, University of Amsterdam), John Kingston (AIAI, University of Edinburgh), Elizabeth Post (Lincoln University), Prof Richard Weston and Paul Gilders (Manufacturing Systems Integration Research Institute, University of Loughborough). Prof Glen Lowry (Convenor, Information Systems Research Group, University of Tasmania), Prof Peter Bernus (Griffith University, vice-chairman of the IFIP/IFAC Task Force on Architectures for Enterprise Integration) and Prof Andy Bytheway (UWC) helped me scope this research, provided valuable high-level direction and shared their invaluable experience. Constructive criticism was also received from Prof. Trevor Wegner (UCT), Mike Uschold (AIAI, University of Edinburgh) and Jean-Louis Van Belle (University of Antwerp).

I also wish to thank the following organisations who helped fund my research travel expenses and conference attendance: UCT, UWC, and the CSD/NRF. Both UCT and UWC granted me six months paid sabbaticals which enabled me to pursue and finalize my research and the CSD/NRF helped fund some of my research and conference expenses. I also owe an intellectual debt to the many conference organisers and delegates who attended my conference presentations and gave valuable constructive feedback.

Finally thanks to three special friends. Jean-Louis Van Belle, motivated me by challenging me to finish before him (he owes me a Belgian beer). Dr Stephen Craven did an excellent proofreading job, provided unlimited homebrew and served as a living example that it is possible to do your thesis part-time. Last, but definitely not least, Caroline Brawner helped with some of the data entry, revitalized my energies by dragging me off to many mountain runs, proofread the entire thesis and provide some valuable criticism.

Table of Contents

List of Tables

List of Figures

List of Abbreviations Used

3D	Three-dimensional
AI	Artificial Intelligence
AIAI	Artificial Intelligence Applications Instituted (University of Edinburgh)
ARIS	Architecture of Integrated Information Systems
ARRI	Automation & Robotics Research Institute (University of Texas)
BOM	Bill of Materials
BPI	Business Process Improvement
BPR	Business Process Reengineering
CAME	Computer Aided Method Engineering
CASE	Computer Assisted/Aided Software/Systems Engineering
CDIF	CASE Data Interchange Format
CIM	Computer Integrated Manufacturing
CIMOSA	Computer Integrated Manufacturing Open Systems Architecture
CIO	Chief Information Officer
CML	Conceptual Modelling Language
CMM	Capability Maturity Model
CODASYL	Conference on Data Systems Languages
CSD	Centre for Science Development
DBMS	Database Management System
DEM	Dynamic Enterprise Modelling
DFD	Data Flow Diagram
DP	Data Processing
DSS	Decision Support Systems
DW	Data Warehouse
EDI	Electronic Data Interchange
EE	Enterprise Engineering
EI	Enterprise Integration
EIS	Executive Information Systems
ERD	Entity Relationship Diagram
ERP	Enterprise Resource Planning
FOL	First-order Logic
GERAM	Generic Enterprise Reference Architecture and Methodology
GST	General Systems Theory
HTML	Hypertext Mark-up Language
ICAM	Integrated Computer Aided Manufacturing
IDE	Integrated Development Environment
IDEF	ICAM Definition (Methodology)
IE	Information Engineering
IFIP	International Federation for Information Processing
IFW	Information Framework
IS	Information System(s)
ISA	Information Systems Architecture
ISO	International Standards Organisation
IT	Information Technology
KIF	Knowledge Interchange Format
KR	Knowledge Representation
MEMO	Multi Perspective Enterprise Modelling
MOF	Meta Object(s) Facility
MOT	Model-Oriented Technology
NRF	National Research Foundation
OA&D	Object-oriented Analysis and Design
OAG	Open Applications Group
OCL	Object Constraint Language
ODP-RM	Open Distributed Processing Reference Model
OIL	Ontology Inference Layer
OMG	Object Management Group

OO	Object-oriented or Object-orientation
OODBMS	Object-oriented DBMS
OS	Operating Systems
QA	Quality Assurance
QC	Quality Control
PERA	Purdue Reference Architecture
PERT	Program Evaluation and Review Technique
RAD	Rapid Application Development
RDBMS	Relational DBMS
SADT	Structured Analysis and Design Technique
SDLC	System Development Life Cycle
SE	Systems Engineering
SEI	Software Engineering Institute
SSM	Soft Systems Methodology
STEP	STandard for the Exchange of Product model data
TOVE	TOronto Virtual Enterprise
UCT	University of Cape Town
UML	Unified Modelling Language
UWC	University of the Western Cape
VSM	Viable System Model
WWW	World-wide Web
XMI	XML Metadata Interchange
XML	Extensible Mark-up Language
XOL	XML-based Ontology exchange Language

Model Acronyms

Code	Short name	Full name
AI	AIAI	The Enterprise Ontology
AK	AKMA	AKMA's Generic DataFrame
AR	ARRI	ARRI's Small Integrated Manufacturing Enterprise Model
BA	BAAN	The Baan IV DEM Business Reference Model
BE	BelgAcc	The Belgium Accounting Framework
BO	BOMA	Chris Marshall's BOMA Model
CY	CYC	The Cyc Ontology
FO	Fowler	Fowler's Object-oriented Analysis Patterns
HA	Hay	Hay's Data Model Patterns
IN	Inmon	Bill Inmon's High and Mid-level Data Models
MI	Miller	J.G. Miller's General Living Systems Model
NH	NHS	NHS Generic HCM Class Model
NI	Nippon	The Japanese Industrial Automation System Model
OB	Ottowa-Big	Ottowa's Business Dictionary Hyperlinks Model
OD	Ottowa-Dense	Ottowa's Business Dictionary Denser Hyperlinks Model
PU	Purdue	The Purdue Reference Model for CIM
RA	Random	A fully random model
SA	SAP	The SAP R/3 Reference Model
SR	Semi-Random	A model with business entities and random relationships
SF	SanFran	San Francisco Application Business Components
SI	Silverston	Silverston et al's Universal Data Models
TO	TOVE	The TOVE ("Toronto Virtual Enterprise") Ontologies
US	USB	The U.S.B. Growth Model

Dictionaries used for semantic analysis

2 letter code	Full dictionary name
BL	BLC or Business Language Corpus
MW	MoneyWords
OB	Ottawa Journal Business Dictionary
SB	Dictionary of Small Business
WP	Washington Post Business Glossary

Chapter 1: Introduction

1.1 Historical Perspective

During its relatively short but momentous half-century history, the discipline of computer-based business information systems has seen more than its fair share of paradigm shifts. This is reflected in the name changes of the discipline: from *Electronic Data Processing* via *Business Computing* to *Information Systems*. For outsiders, these paradigm shifts were characterized by changes in underlying technologies: hardware went through different generations of computers; network architectures evolved from centralized to distributed systems; and software applications progressed from stand-alone functional batch transaction processing systems to interactive, DBMS-driven enterprise-wide applications. Commensurate with these high-visibility shifts, more subtle but equally important changes occurred in the management of the business IT function (from DP manager to CIO), skill sets of IT staff, the nature of IT projects and the role of IS in the organization at large.

Of greater concern to this study, however, is the evolution in the development of information systems: the move from low-level and structured programming languages to DBMS query-languages, program generators, visual programming and CASE tools. This necessitated a shift from programming centred development (originally "seat-of-the-pants", later "structured") to the discipline of systems engineering and the design of SDLC methodologies. One of the keys elements in this shift was the development and growing importance of domain analysis and requirements specification activities. This in turn sparked off the development of a whole slew of domain modelling technologies, of which the graphical modelling techniques are more prominent in the business arena. The more popular techniques are the Data Flow Diagrams *(DFD)* and Entity-Relationship Diagrams *(ERD)* from the relational database era and the Unified Modelling Language *(UML)* as an exponent of the Object-Oriented *(OO)* age. The recent trends in the systems development area are perhaps best characterized by the term *model-driven* development whereby the development of most large business information systems starts with a domain analysis model. And, as more and more of the programming and implementation activities are automated, the importance of the modelling phase grows.

The move from functional applications to enterprise-wide applications, and the shift from a functional (vertical) to a process (horizontal) approach using the value chain, has necessitated an increase in the scope of the domain models to integrated enterprise-wide models. This necessity is further amplified by the desire to build data warehouses *(DW)* to satisfy the hunger of functional managers for integrated information as well as the desire of IS managers to develop long term strategic plans based on, *inter alia*, an enterprise information architecture. Modelling in general has become serious business, and enterprise modelling is seizing a significant and growing piece of the action.

Academics and practitioners alike have been quick and prolific in developing different development tools and methods. Sometimes, it appears as if there are as many methodologies as there are academics and consultants. Consequently, there have been numerous studies, frameworks, evaluations and comparisons of development methodologies and tools (programming languages, modelling notations, CASE tools, IDEs). However necessary these studies of the relative merits of various development tools are, it is the *output* or *product* which is important from a business point of view. The purchaser or end-user of a system does not care whether it was written in C++ or assembler, whether a structured programming or object-orientation approach was used, whether the data is stored in a RDBMS or an OODBMS, etc. The user will perform a system evaluation based on the quality of

the various emergent properties of the systems such as user-friendliness, scalability, response time, flexibility or cost. At this point, there seems to be a particular dearth of guidance available on how to evaluate the actual output of the modelling activity: what measures or techniques exist to evaluate an enterprise model? Since the quality of a model will have a significant impact on the IS function, regardless of whether the model is used as an enterprise data standard, the blueprint for a strategic information architecture, the meta-model for a data warehouse repository or the actual development of an information system (component). The quality of a model is not necessarily dependent on the actual methodology, tool or notation that was used to build the model, though it may, of course, be a strongly influencing factor.

1.2 Purpose/Scope of the research

The purpose of this study is ***the development and validation of a comprehensive framework for the analysis and evaluation of enterprise models***. It is important to elaborate on each element.

- "**Development**": no comprehensive framework has been found in the literature although a number of frameworks for the evaluation of other "intellectual products" exist. These are used as inputs to guide the building of an integrated and comprehensive framework.
- "**Validation**": The development of a theoretical framework does not contribute towards science unless it is accompanied by a verifiable and methodological testing and evaluation of the framework itself. A "test bed" will be constructed, consisting of a substantial number of publicly available enterprise models from different reference disciplines and from both commercial and academic origins. The database containing these enterprise models in a common format, XML, will form a substantial contribution in itself to researchers in the information systems discipline.
- "**Comprehensive**": although many independent measures and metrics for model evaluation exist, these have usually been presented in relative isolation. An integrated framework to include measures from many perspectives is necessary to relate these metrics to each other.
- "**Framework**": no attempt is made to formulate a set of procedures or methodologies for applying the framework, though guidelines will be provided for interpretation of the various analysis results on their own, related to each other and in context.
- "**Analysis**": a number of metrics and measures already exist, others will be imported and/or adapted from other, related fields and a few new ones will be developed and tested.
- "**Evaluation**": there are intrinsic qualities (absolute measures that can be computed for one specific model) and comparative qualities (relative measures that compare models). Some of these entail a ranking or judgement (better, worse) whereas other measures merely differentiate (e.g. model A is more like model B, whereas models C, D and E form a separate family).
- "**Models**": in principle the framework should be formulated and validated in the context of any possible enterprise model. Because of practical and methodological reasons (as explained in Chapter 3) the emphasis will be on static models. Extensions will be suggested on occasion to cater for dynamic models, although it should be recognized that this is not necessarily a trivial exercise.
- "**Enterprise**": On the other hand, it should be fairly trivial to use the framework for the evaluation of models in other domains (e.g. to engineering applications by changing the reference semantic corpus for the semantic analysis) or even sub-domains within the enterprise (e.g. specific functional areas).

1.3 Scientific Contribution and Importance of This Research

The research in this thesis makes a number of significant contributions, both to the Information Systems discipline as a science and to practitioners in the field. The following gives an overview of the relevance of this research to the IS community.

The enterprise **model evaluation framework** is an original scientific construct, filling the gap between the many frameworks for the evaluation of development tools and methodologies on the one hand, and the literature on the evaluation of IT software and hardware products on the other. Although the framework is not presumed to be the definitive and final word on model evaluation, it is believed to be specific and comprehensive enough to withstand a first round of academic criticism, yet extendible enough to invite further elaboration and refinements by fellow researchers. In addition, the framework is specific and clear enough to be implemented without further development by decision makers wanting to obtain off-the-shelf models. It can also serve as a guide to modellers and/or their managers wanting to evaluate their own in-house model(s).

In the course of operationalizing the framework, **metrics and measures** have to be developed for each aspect of the framework. Although several are imported from other disciplines, a number of brand new ones have been devised, and many more adapted to a greater or lesser extent to suit the particular domain. These measures form a valuable intellectual contribution in their own right to the field of systems engineering metrics and the modelling discipline, and can be implemented or evaluated regardless of the overall framework. In the field, these measures will prove useful for system development tool builders (e.g. for use in upper-CASE tools) or for IT project managers and modellers to judge productivity and/or quality.

The research uses 'off-the-shelf" generic models for the validation of the framework i.e. models that are publicly or commercially available. This is believed to be the **first scientific comparative evaluation** of these models in the literature. This evaluation should provide useful feedback to the builders of the models and to possible future developers of new enterprise models. A particular focus of this research is its interdisciplinary approach, drawing on models from a wide range of reference disciplines.

The comparative evaluation requires the capture and standardization of a substantial number of enterprise models. The resultant **database** with these models in XML format will prove useful for future research. Although the public availability of some of its specific contents may be constrained by the intellectual rights pertaining to some of the models, a partially populated database can be supplied, which includes the full syntactic and most of the semantic data. Almost equally important to the content of the model database is its internal architecture and implementation of the **meta-model**.

Finally, the **interdisciplinary** character of the research should provide some stimulating impetus and serve as a fresh breeze to anyone working in the field. It is not often that outputs from such a wide variety of disciplines are brought together: enterprise models originate from such diverse sources as systems theory, computer science, ERP, accounting, linguistics and systems engineering. The development of the framework itself even incorporates contributions from such diverse sources as construction architecture, complexity theory and aesthetics.

1.4 Structure of This Thesis

The literature survey has been split into two separate chapters. Chapter 2 deals with the literature relating to the concept of enterprise modelling and establishes the terminology and conceptual framework. Chapter 3 investigates in more detail the various reference disciplines and more specialised research areas in which enterprise modelling is pursued and from which the enterprise models are drawn. The latter is important because it determines the context and methodological bias inherent in the different generic models. In both chapters, a conscious decision was made to be as comprehensive as possible (refer to the extensive list of references) as well as to include sufficient detail for those readers who may not be equally well-versed in the various disciplines or may be lacking in some of the academic background. Readers familiar with the reference disciplines may want to skim read these sections.

Chapter 4 documents the methodology which has been adopted for the development and validation of the framework. Chapter 5 contains the proposed framework, but also includes an overview of the relevant literature from which the framework's constituent elements were drawn. Chapter 6 discusses the database of sampled enterprise models as well as providing some notes on the capturing process and the XML format in which the database is made available to other researchers.

Chapters 7 to 9 are concerned with the operationalization and validation of the proposed framework according to the three main dimensions of the framework: syntactic, semantic and pragmatic analysis criteria. Chapter 10 discusses a number of analysis techniques which were investigated but could not be adopted due to methodological limitations. It also suggests other IS research areas where the framework could be used. Chapter 11 contains the overall conclusions, including a summary of the main findings, and areas amenable to further research.

The thesis concludes with a comprehensive list of references to the literature that was consulted for this thesis. This is followed by a number of appendices which contain details that would have interrupted the flow of thought if they had been included in the main body of the thesis. They contain illustrative samples of the raw form of the various enterprise models, some of the conversion issues that were faced and more detailed or secondary calculations and evaluations

Chapter 2: Models

Chapters 2 and 3 of the thesis build the scientific context for the thesis by means of a comprehensive literature survey. Since it is hoped that some of this material can be used for textbook and/or course curriculum purposes, the aim is to have a somewhat greater level of detail than is normally found in doctoral theses. This chapter deals with the current state of enterprise modelling. The next chapter discusses the various reference disciplines contributing to the frameworks relevant to the thesis.

This chapter explores the basic terminology and scientific foundations of enterprise models: what a model is, model uses, types of models, the role of modelling in information systems, enterprise models, model representation and meta-modelling. The latter two topics are of great importance to the development of the framework and are therefore discussed in greater detail.

2.1 What Are Models?

Modelling is done in many disciplines, by many researchers from different philosophical backgrounds and in many different contexts. Hence the word *model* has acquired several meanings and definitions, resulting in what some authors refer to as the "model muddle" [EDMO99].

For purposes of this research, the following definition will be adopted:

> "A model is an abstract representation of reality that excludes details of the world which are not of interest to the modeller or the ultimate users of the model." [PRES97b]

In principle it is almost possible to use anything as a model of anything else; e.g. "cutlery on a table may represent the disposition of troops at the Battle of Waterloo" [ROBI97]. The "reality" that is modelled is usually referred to as the *object domain* and may be physical (e.g. the solar system), abstract (e.g. modelling social relationships in a group) or a combination (e.g. an organization which consists of physical artefacts, legal entities, individuals, structure etc.). In most sciences, informational constructs rather than physical representations (such as a scale model) are used to build the model. These constructs are then represented themselves by means of symbols, e.g. a UML diagram. This creates a triad: from the real world (*domain*) one builds (by means of perception and conceptualization) the conceptual model and, by means of signs or language, finally creates a *representation* of the model [SALT93]. This is illustrated in the *meaning triangle* (Figure 2-1) which is adapted and elaborated from [SOWA00].

Note that the conceptual model V2 reflects the model semantics (or meaning); and the model representation V3 emphasizes the model syntax (structure). In practice, when referring to "the" model, the amalgam of both conceptualization and representation is generally implied. Sometimes, a stricter interpretation is used, as in Presley's definition which refers to V3 only.

The meaning triangle can be a useful framework to position the various disciplines dealing with modelling (most of these will be discussed in the next chapter). Consider the following:

- Methodology engineering will look more closely at building frameworks (modelling) of the processes or activities between the vertices V1 to V3, as well as at what form the representation should take.
- Meta-modelling can be seen as adding a second "meaning triangle" to the right of the first triangle, by taking the symbolic model representation as its object domain i.e. it models the modelling constructs used in the model representation: V3 of the first triangle is V1 of the second. It is possible to add a third triangle, as is done in meta-meta-modelling (see 2.11.4).

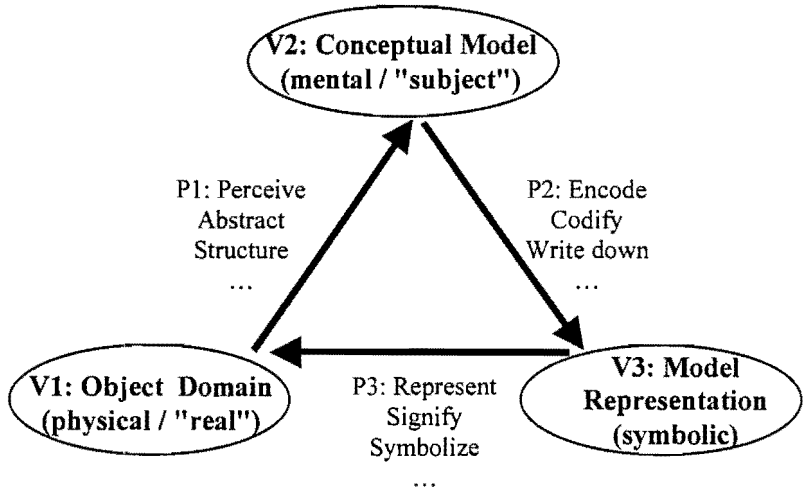


Figure 2-1: The Meaning Triangle, adapted from [SOWA00].

- The executable models used in enterprise integration blur the distinction between the represented model V3 and reality V1 i.e. the model becomes an intricate part of the object domain and V3 is conflated into V1.

Some general notes about modelling can also be illustrated by means of specific elements of the meaning triangle. The following deserve some further discussion:

- V1: The real domain/object. A real object does not need to be material, and can even be fictitious. Although some objects are purely physical (a river, the physical world, a molecule, a virus), many objects modelled in the social sciences include conceptual or non-material elements: the economy, an enterprise, or a war. Similarly, an object does not necessarily need to exist to be modelled: virtual reality games model non-existing castles, life forms and planets; toy shops sell plastic models of the starship Enterprise.
- V3: Model representation. Engineers and architects often use physical models for V3 e.g. a small-scale model of a building. One can even construct a physical model of a conceptual object domain such as the "coloured-fluids-in-pipes model" (physical V3) of the Dutch economy (abstract V1).
- P1: The abstraction relationship. This conscious process is referred to as modelling the object. The required degree of fidelity between V2 and V1 depends on the purpose of the model: where a highly abstract, high-level view is required, fidelity (or accuracy) is not necessarily desirable. There are various methodologies and guidelines available, depending on the discipline. For instance, mathematical modelling of physical phenomena as opposed to, For example, the theoretical models of socio-economic objects. The modelling process actually constitutes the main research body of entire disciplines such as statistics, the cognitive sciences or econometrics.
- P2: The encoding relationship. Ideally, V3 is as close as possible to V2. Deductive models, as e.g. developed in ontology research, aim to integrate the conceptual model V2 (semantics) and representation layers V3 i.e. a modelling notation with extremely rich semantics is used.
- P3: The representation relationship. The model is not an exact duplicate of the real object, which implies a partial fit or projection relationship. Many of the model's qualitative attributes refer to this relationship: simplicity, accuracy, completeness and time lag. P3 is a bi-directional relationship. The modeller represents his/her mental model V2 by means of V3. The representation V3 then serves as a communication tool for the reader to "reconstruct" the mental model V2 in the reader's mind, hopefully as close as possible to the mental model V2 held by the

modeller. A close correspondence between V1 and V3 lessens the cognitive effort required in reconstructing the mental model V2 e.g. in a scale model, photograph or graphic film scene.

Modelling is a natural human activity by which we make sense of the world around us [DEME82]. Without building mental (internal) models of the world, we could not function: we expect a hammer to fall when we release it and a ball to roll when we kick it. We know that a container can hold water but a flat surface will not. We expect the sun to rise in the morning and we know what to do when we walk into a restaurant. However, this does not imply that our models are necessarily correct (does the sun revolve around the earth?) or that we share common models (what a Roman Catholic sees as "a priest celebrating Mass" was described by Sartre as "a man drinking wine in front of women on their knees").

Scientists construct models of complex phenomena in an attempt to understand or predict the "real world". In the social sciences, these *models* are akin to *theories*, in the sense of a model of human cognitive visual perception, for example. By contrast, a clear distinction between theories and models is drawn in the physical sciences [EDMO99]. In the information and business sciences, models are used for analysis, design and decision making, and hence geared at presenting the model user with the information necessary for that task, while omitting other irrelevant or detail information [WHIT98d].

Models range from simple (Newton's law of gravitational force) to complex (an econometric input/output analysis model of a country's economy). [BEER99] points out that models are necessarily limited because of the limits of our sensations or perceptions (an epistemological limitation) and subjective "because individuals differ in acuity of perception and in pattern penetration". In addition, reality is often richer than the number and variety of constructs used in building the model i.e. the *expressiveness* of the modelling language limits the model [SALT93].

Often overlooked, but quite important when discussing generic enterprise models, is the distinction between a model of an object and of the class of those objects. In some cases, the distinction is vague because the model of the class can easily be instantiated to apply to a specific case. For instance, a general macro-economic model can be tested and applied to a specific country's economy by estimating the parameters for the given country, though the estimation is not necessarily a trivial exercise. On the other hand, an organization theorist may build a theoretical model of organizations in general e.g. how organizational structure affects communication efficiency. The model of the organization theorist is probably useless to anyone wishing to analyse a specific organization because ofits highly abstract nature. A slightly finer distinction is found in physics: physicists are not interested in modelling a specific physical mass, molecule, atom or fundamental particle, just the generic molecule. Luckily for the physicist, instances of fundamental particles are perfectly interchangeable. When dealing with more complex objects, we have the additional burden of the variability between class instances e.g. not every patient's body will react in the same way to a drug, not every group of first year students is the same as another one of similar size etc.

2.2 Model Attributes

Inasmuch as a model is about another object or entity, it shares many of the attributes of any information or data item [VANB99a]:

- **Accuracy** or fidelity: how well does the model represent its object? Although the final accuracy of, say, a prediction is a function of model accuracy, completeness and reliability, the accuracy attribute refers to the intrinsic variables and functions within the confines of the model.

- **Reliability**: how dependable is the abstraction? Often an important criterion for the predictive capacity, it refers more to the process by which the abstraction was conducted. Even if one uses a statistically large enough sampling, the sampling frame may be inappropriate.
- **Completeness**: how many of the domain objects' attributes, dimensions or views are incorporated in the model?
- **Verifiability**: to what extent can the correctness or validity of the model be checked? A model needs to contain variables which can be checked through direct observation (empirical verification) i.e. compared with the real world object.
- **Validity**: how well does the model correspond to the real world? This issue is expanded below.
- **Relevance**: how pertinent is the model with respect to the context of the use for which the model was developed? Models are often unfairly criticized as inaccurate or too simplistic without taking into consideration the original motivation for the development of the model.
- **Simplicity**: how complex is the model? This refers to the number of dimensions or attributes that has been incorporated, and to the relationships between them. A simpler model is easier to comprehend, communicate and verify but may be less accurate. This dimension is almost but not completely the opposite of completeness.
- **Compactness**: How compressed is the model compared to the domain object? Fewer model elements often imply a more compact model but not always: the classic Newtonian model of the physical universe was simpler but Einstein's more complex general relativity model more compact because it combines several different sets of laws into one set.
- **Timeliness**: what is the time lag between changes in the underlying object and the corresponding model? Modelling the global weather to a great degree of accuracy could be slower than waiting for the weather to happen; the time lag is due to the model complexity itself. Modelling the economy suffers from the problem of time lags in measuring some of its variables; as does the accounting system in many organizations.
- **Cost**: how many resources need(ed) to be invested to develop and maintain the model? This refers primarily to financial and human resources, but can be extended to computational (algorithmic and storage) requirements.

Note that these attributes are not fully orthogonal e.g. model cost correlates (to a degree) with accuracy, timeliness with relevance, completeness with reliability and compactness with simplicity.

2.3 The Philosophical Bias of Models

The presence of a modeller guarantees that there is no such thing as an absolute, perfect model. All modelling remains a subjective activity since the modeller, because of her *Weltanshauung*, introduces bias into the model.

There are two extreme viewpoints. The subjectivists will argue that much of what is called "reality" (V1 is in the meaning triangle) is really a social construct and hence all models of complex phenomena are intrinsically subjective. At the opposite end of the spectrum is the naïve realist claiming the existence of an observer-independent physical world which can be perceived and modelled to whatever degree of accuracy [HIRS89].

Constructivism is very prominent in the social sciences whereas physical scientists tend to have realist leanings. Thus this philosophical debate rages especially strong in IS, which draws its roots from both disciplines. The soft systems methodologists (*SSM*) claim that:

> "models are not would-be descriptions of parts of the world. They are abstract logical machines for pursuing a purpose, defined in terms of declared worldviews, which can generate insightful debate when set against actual would-be purposeful action in the real world" [CHEC92].

This is in sharp contrast to the often unspoken conviction of many systems engineers (usually of the so-called "hard modelling" school), who may believe that there *is* such a thing as a best or most accurate domain model [FORB95].

The position taken in this research is somewhere in the middle of the spectrum, though with strong Platonic-realist leanings. As [FLYN96] points out:

> "organizations [are] socially constructed phenomena, whose many aspects may be perceived differently by different observers, but which are able, by negotiation, discussion or some other method, to formulate a common view for a period of time, containing a minimum of inconsistencies."

There seem to be sufficient commonalties across organizations and the members of the social groups studying these, that a large number of generalities, common perceptions and collective vocabulary has been established. The use of different model views (see below) can, to a certain extent, help reduce the subjective bias in models. The *realist's* view adopted here, especially in respect of data models, is strengthened by Avison and Fitzgerald's "admission" that the theoretical subjectivist arguments may not necessarily apply in practice:

> "The assumption in data analysis that it is possible to model reality is questionable [...] The data model can only be *a* model and not *the* model of that part of the real world being investigated. It cannot reflect reality completely and accurately for all purposes. Even if data analysis has 'gone according to plan', the resultant data model cannot objectively represent the organization. It is a subjective view distorted by the perceptive process. *Having said this, however, the data model derived from data analysis usually proves in practice to be suitable for the purpose of building a database.*" [AVIS95]

2.4 Why Do We Model in Information Systems?

Before building a sizeable information system, it is necessary to develop a model; much like a blueprint is needed for the construction of a large building.

The IS model serves a number of different purposes [WHIT98d, PRES97, LAND87]:

- Modelling assists in the analysis of the enterprise and the subsequent *design* of the physical systems.
- It helps the *understanding* of a complex system; in part because it reduces the complexity by breaking the system into smaller pieces.
- It allows one to *communicate* a common understanding.
- It can be used to obtain *stakeholder buy-in*, especially if all interest groups have a say in building the model.
- It can serve as a *documentation* mechanism e.g. for ISO9000, quality control etc.
- Some types of models provide *simulation and forecasting decision support*. These can assist decision makers in controlling, predicting and optimizing decisions.

The increasing scope and complexity of information systems (such as data warehouses, ERPs and on-line distributed applications), and developments in system development methodologies (OO, CASE,

IDEs, UML), are driving the transition from code-centred to model-centred systems development [BEZI99], who also claims that this is the precursor to the paradigm shift from OOT to MOT (model-oriented technology) [BEZI98a].

2.5 Types of Models

There are many typologies for the classification of models. Edmonds [1999] lists a large number of generic classification schemes, e.g. according to the medium in which they are expressed: physical (a scale model of an aeroplane), mathematical (structural equations model), computational (a computer program) or linguistic (using natural language). After listing many more, he concludes that it is impossible to make a comprehensive list of model typologies.

[LYYT87] distinguishes the following types, depending on the representation used to describe the model: iconic (looks like the reality it is intending to represent, e.g. a scale model); analogue (similar in relations but using different entities e.g. a map); symbolic (uses abstract symbols e.g. a decision-making model); schematic (uses a diagram or chart to represent the model state); mathematical (uses mathematical symbols e.g. equation model); verbal (English description) or conceptual (theoretical explanation).

The following are some model typologies of relevance to *information systems* models:

- Conceptual versus data model. The conceptual model captures all relevant static and dynamic aspects, i.e. all rules and laws etc. relating to the domain. The data model describes what *data* should be captured by the information system [OLLE93].
- Viewable and executable models. Some models are used for communicating views of the system during the analysis and design stage of an IS whereas others, e.g. those expressed in programming language or similar formal notations, are directly executable by the computer [VANH99].
- Active and passive models. Once a passive model is created, it is independent of its subject (or domain). Subsequent changes in the domain are not (automatically) reflected the model. This is typically the case for most system development methodologies. In an active model, the relationship between the model and its subject is maintained so that the model reflects the current state of its subject. This synchronization, typical of control systems, does not have to be immediate [GREE95]. An active model is similar to the "living model" of [WHIT97a].
- Static or dynamic models. Static models give a static representation of a usually dynamic system e.g. the types of objects in the system and their flow paths through the system. Dynamic representations describe the behavioural aspects of the system and are typically used in a predictive manner e.g. by means of "what-if" scenario analysis [WHIT98a; WHIT98b].

2.6 Views and Layers in Models

2.6.1 Views: Modelling from Different Perspectives

Modelling the real world involves abstraction and subjective perspective. Ask a farmer, an economist, a biologist, an artist and a chef to describe a strawberry and each will most likely produce a radically different description. Or, in an enterprise context, consider the widely different conceptualizations of a production process as made by the shop floor worker, cost accountant, production manager, PERT analyst, maintenance engineer, psychologist, Marxist philosopher and BPR consultant. In order to describe an organization from as many perspectives and as completely as possible, models adopt the

structuring concept of model *views*. Views accommodate the different needs of the model users, as well as the different types of information available about the enterprise.

Model views are based on subsets of modelling concepts:

> "A single model [...] would be overwhelming in complexity. Therefore, we need a way to separate concerns such that we can check consistency between alternate specifications of the same system." [FARO97]

Hence, the viewpoint concept - each viewpoint looks at the whole system, but uses modelling concepts specific to a defined subset of modelling concerns. The different viewpoints are not necessarily different levels or layers in the model, but different, often complementary, aspects or abstractions of the same model. This means that a single concept in one view may be represented by multiple concepts in another view. Views must have some overlapping modelling concepts to enable users to relate the models to each other. If the common ground between the views is too small, it may create cognitive problems.

Exactly how many views are necessary for modelling organizations is not evident [FRAN99a] and almost appears to be a matter of academic taste. The following serves purely to give a flavour of the breadth in the number and types of views that are possible. Modellers in the early days of programming used process flowcharts only. Relational database modelling often restricts itself to two views: dataflow and entity-relationship modelling. The ARIS (Architecture of Integrated Information Systems) methodology [SCHE94, SCHE98, SCHE99] uses four viewpoints: data, function and organization as the three fundamental views and a fourth view which is the resource view when modelling the information systems or the control view in the business system context. These views are similar in nature (though slightly different in implementation) to those defined in the CIMOSA (Computer Integrated Manufacturing Open Systems Architecture) framework [BERN96a]. The ARRI (Automation & Robotics Research Institute) adopted five views: resource, activity, process, organizational and business rule [WHIT98c]; though they are very different to the five used by the ODP-RM (Open Distributed Processing Reference Model): enterprise, information, computational, engineering and technology. The Zachman framework [ZACH87, SOWA92] requires that six "dimensions" be considered: data, process, network, people, time and purpose. Bubenko proposes the use of eight interrelated sub-models or views to model the enterprise: objectives (why), concepts (what), activities (how), actors (who), functional and non-functional requirements, configuration and information system [BUBE00]. In the WORKS approach, which is based on the CommonKads knowledge-based systems modelling, nine views are introduced: organization structure, organization processes, staff, working tools, data view, communication and cooperation, expertise, (knowledge) sources, and strength/weakness view. [DECK97]. The UML (Unified Modelling Language) allows for at least ten different types of diagrams: use-case, class, object, sequence, collaboration, statechart, activity, component, deployment, and the package diagram [BOOC99, HALP99], though no real-world methodology uses all ten UML diagram types simultaneously for any given situation.

Note that the discussion above refers to conceptual views. In database theory, another hierarchical set of views or perspectives called "*schema*" is introduced. Based on the CODASYL framework, [HAY00] mentions the following four perspectives in reference to corporate data models:

- The *conceptual* schema representing the organizational or "full" view of the data.
- The *external* schema being the individual user's view of the data, depending on the user's function or data needs. This is a subset of the conceptual schema.

- The *logical* schema represents the view of the DBMS of the data, typically in terms of tables, columns, network segments etc.
- The *internal* schema represents the physical data structure as stored internally on the computer system.

For purposes of this research, only the conceptual schema or perspective will be of relevance when discussing modelling views, unless indicated otherwise.

2.6.2 Layers: Modelling at Different Levels of Detail or for Different Purposes

An equally important issue is to model, not only from different viewpoints, but also at the appropriate level of detail. Ideally, a model should be available at different levels or layers of detail, by means of an "explosion" or "zoom" facility, to increase or decrease the scope view of the model. Some of the literature refers, confusingly, to these layers as views, whereas, in fact, model detail level is orthogonal to the views dimension. This means that a model may simultaneously have a number of different views, each at different levels. Sometimes, the system is not modelled in more detail, but for a different type of user/purpose, e.g. a shift from conceptual to implementation. Although the latter "layer" will require more detail, there is also a shift in emphasis or purpose. This may lead to differences in the model that are not just a matter of detail.

Systems engineering has traditionally recognized two different scopes in its models: the business model as opposed to the system model, although the latter can then be further specified into analysis and design or, as in ARIS, requirements, design and implementation models. MEMO (Multi Perspective Enterprise Modelling) differentiates between three levels (confusingly referred to as "perspectives"): strategy, organization and information, though each perspective needs to be structured according to four aspects (structure, process, resources and goals), thus requiring a total of 3 x 4 = 12 foci to be considered [FRAN99a]. CIMOSA proposes three different levels: generic, partial and particular (each structured in the four views organization, resource, information and function), leading to 12 cells. Zachman's six dimensions are to be modelled at six different levels (ballpark view, owner's view, designer's view, builder's view, detailed representation and functioning system), creating a potential 36 different sets of models [COOK96].

2.6.3 Problems with Views and Levels

The only consensus in the "methodology jungle" is that there should be at least two views: a static data or information-oriented view, and a dynamic activity or process-change view. Similarly, there is widespread agreement that there needs to be different levels of models, at least for sufficiently complex domains, ranging from the high-level overall view of the enterprise which is technology and system independent, down to a much more detailed low-level technical implementation. However, there is still no consensus on the kinds of views, nor on the number of layers of detail required.

The use of views and levels results in a key issue in modelling theory: the integration of these different levels and views. Although correspondence can be enforced to a certain extent by the modelling tools, a number of practical and conceptual problems remain [FRAN99a]. Whitman identifies four key issues with respect to the synthesis of views [WHIT98d]:

- **Potential gaps in view**: since each view omits certain aspects, any single view is not sufficient for an analysis of the domain. For instance, to model a process in UML requires either extending (customizing) one type of diagram or bringing together the information contained in several different diagrams.

- **Artificial wrappers**: in order to populate higher or intermediate levels of enterprise models, some abstract entities (which are to be broken down in more detail at the lower levels) are often necessary, but these are not necessarily readily recognized by individuals familiar with the real domain. An example could be the failure of a line employee to understand the concept of "party role" (representing both customer and supplier), or the term "asset" to describe liquid, fixed, current, human and informational resources.
- **Differences in structure**: some views may allow certain structures which cannot be accommodated easily or naturally in other structures. Both activity and process views facilitate hierarchical decomposition more naturally than the data view.
- **Model ambiguities**: concepts that are easy to represent in one view of the model may require additional information, different or additional constructs, or artificial decomposition to allow representation in other views.

Another important problem is that of context: within an organization, different departments may refer to different concepts by the same name, or use different names for the same concept [HICK99]. Many of these issues have not been resolved by the current generations of modelling and CASE tools, as discussed more fully in [HAY00].

2.7 The Process of Modelling

Despite the best efforts and claims of methodology and systems engineers, modelling remains as much an art as a science. Modelling a domain requires a combination of aptitude, training and experience. Many guidelines exist to assist the aspirant modeller, but the modelling process is often a subjective, personal experience.

2.7.1 Formalizing the Modelling Process

Methodologies attempt to structure the system development process, of which the modelling activity forms an important part. A methodology generally prescribes a set of steps or activities, along with the adoption of specific modelling tools or representations. Many large organizations, and IT consulting companies in particular, will develop or enforce their own brand of systems development methodology, which will lay down specific modelling activities, deliverables and notations. Other organizations may adopt off-the-shelf methodologies, often in conjunction with a specific tool e.g. the Rational Process with Rational Rose for UML [BOOC99].

The main purpose of a methodology is to obtain consistency and maintain minimum quality standards. This is often linked to project management, metrics collection and quality assurance programs. Where quality assurance programs are to be formally accredited, much heavier demands are placed and the degree of formalization increases dramatically, as is the case for e.g. ISO9000-3 compliance or, for a model more specifically adapted to information systems environment, the CMM (Capability Maturity Model) of the SEI (Software Engineering Institute). The latter specifies five levels to characterize the maturity of an organization's software development process. Already from the second level there is the requirement to have formal policies in place to ensure repeatable processes. See [MULL97] for an interesting suggestion on how the CMM can be specifically applied to a data model.

Nevertheless, even expert modellers will often model the same domain in different ways, suggesting that no one optimal model for a given domain exists. Although methodologies are discussed elsewhere, a couple of general issues are worth mentioning.

2.7.2 Iterative Process and Holistic Approach

Many methodologies assume or imply that modelling is a linear activity. In reality, it is an iterative process whereby the domain is explored by means of initial "rough" models, which are gradually refined and amended [LUKO96]. This is particularly advisable when modelling a vague domain [MENZ96]. Unfortunately, a "first-cut" model often tends to stifle novel approaches and conditions the modeller's mind into a set pattern. Frequent interaction with other domain modellers and users is therefore imperative to prevent "straight-jacket thinking", especially in the early stages of the model. This exploratory stage of modelling often benefits from one of the many creative thinking approaches, such as brainstorming sessions. Additionally, it must be acknowledged that modelling a large and complex domain requires a holistic approach:

> "It is common experience that attempts to solve one piece of a problem first, then others, and so on, lead to endless solutions. You no sooner solve one aspect of a thing than another is put out of joint. And when you go back to correct that one, something else goes wrong... This is the great argument against the attempt to solve [modelling] problems piecemeal." [WITT94, p.15]

2.7.3 Similarity of Modelling to Other Development Processes

A number of authors have pointed out the similarity between the model development process and other processes. For example, [WORT00] points out the following five similarities of product development with enterprise model development (and model life cycle management):

- Both require a versioning mechanism.
- It is essential to describe both from different viewpoints (possibly requiring the use of different formalisms), depending on purpose.
- The development status needs to be tracked until the design is satisfactory.
- There needs to be a way to record commonalities and differences if more than one variant of a particular product is to be designed.
- A hierarchy or decomposition mechanism is essential to hide or expose complexity, usually by means of both a zoom in/out facility as well as a "BOM" recursive component type approach. (as discussed in the section on model "levels").

In a similar vein, the Zachman framework was initially derived for the construction engineering sciences using architecture principles [ZACH87], but was then applied to the field of IS architectures.

2.7.4 Some Modelling Principles

Many of the following modelling principles find their origin - or equivalent - in the object-oriented world. For instance, OO models will tend to "inherit or import" an object's intrinsic high cohesion from the real world into the model. The two fundamental principles in the design of complex models are [WITT92]:

- Loose coupling/high cohesion. Cohesion refers to closely related concepts which must be grouped together, often in a single higher-level concept. This allows the use of information hiding or "black boxes" where the detailed, closely interrelated information can be hidden into one single package. The loose coupling refers to the opposite principle i.e. the grouping of concepts in such a way that the relationships between concept chunks are minimized. This has the advantage of relative design/analysis independence, easier conceptual understanding and quicker model evaluation.

- The generous use of abstraction, especially in the high-level design. There are many abstraction mechanisms available, such as looking for common attributes, the creation of complex objects and the identification of strong cohesion or one-to-one relationships.

Other suggestions on how to structure and classify objects rely on the use of structured reference sets wherever possible, e.g. controlled word lists, ontologies, taxonomies, and thesauri [NEWT98].

A final point of note, perhaps better labelled as an "anti-principle", is the fact that modellers often tend to overemphasize the discreteness and static nature of the real world. It must be realized that there are often unclear boundaries between different concepts, a phenomenon known as gradience, fuzziness, impreciseness, vagueness or fluidity [HONK98]. Many "discrete" concepts really refer to a continuum of situations or objects. This is not limited to intrinsically fuzzy concepts such as quality, taste or preference. A common example is the definition of a customer which may or may not include any of the following:

- A prospect who made an enquiry.
- Someone who requested a quote.
- Someone who first accepted a quote verbally but never confirmed the order in writing.
- Someone who placed an order but cancelled it.
- Someone who placed an order for an out-of-stock or unavailable item.
- Someone who placed an order but never received delivery.
- Someone who received a single delivery but returned it.
- Someone who received a delivery but never paid for it.
- Someone who has bought but since moved out of the organization's operations area e.g. someone who emigrated to Japan.
- Someone who has not ordered anything for a "long time" (how long?).
- Someone who ordered in the past but has no reasonable ground to order in the future (due to changes in life style, purchasing power, status etc.) e.g. a company that has decided to manufacture your products in-house.
- A previous customer who has since made a formal decision to switch suppliers to the competition (an important legal or status issue if the customer was the Queen of England or the US Department of Defence).

Related is the tendency of many modellers, especially those from the "hard modelling school", to over-emphasize the *static* aspects of reality. The world is then often described in entities and relationships between entities, losing sight of the more problematic *dynamic* changes in reality which are much more difficult to model. The attempt to attach a single verbal "one-to-one" label to real-world entities that are slowly changing their identity presents more than just philosophical problems. It is perhaps not surprising that some authors suggest that models should *not* start with a domain data model but with user interactions or system goals analysis [KAAS96; LARM98].

2.7.5 Model Validation and Verification

An important activity in the modelling process is that of model verification and validation [CHEN98]. Indeed, this introduces a strong *ethical* dimension: an invalid or incorrect model - especially if it is used for subsequent system design - has the ability to impact very negatively on the profitability and hence survival and growth of an organization, as well as the quality of (work) life of its employees and other stakeholders. Therefore modelling is a serious and responsible activity, and choosing the correct methodology as well as ensuring proper model validation is "as much an ethical consideration as a rational one" [JAYA94].

Model *verification* focuses on the syntactic level: it is the activity that checks that no modelling rule has been violated e.g. illegal constructs, incorrect use of symbols, inconsistencies between different

model views etc. With the increased use of integrated and sophisticated modelling and CASE tools, much if not most of the verification is automated.

Model *validation* is checking that the model is a true (or at least valid) representation of the real world. This involves both semantic (what do the modelled concepts really mean) and pragmatic (will the model be used and interpreted properly) issues. Model validation is often an informal process [BERG00]. Several techniques can be used such as paraphrasing the model in natural language, using graphical tools, simulation of the model, explanation generation techniques, various user-oriented graphical visualization tools and translation into user-defined concepts. Because many of the stakeholders have a limited knowledge of modelling, this process is often not very satisfactory [FLYN96]. As Bergholtz points out, it is difficult even for experienced designers to validate a model.

Indeed, there is a dearth of concrete, usable research findings in the area of model validation. Model validation is a major thrust of this research project and will therefore be discussed in much greater detail in Chapter 5.

2.8 Enterprise Models

The models of interest to this research are enterprise models. This section discusses the nature of, reason for, and the current state of enterprise modelling

2.8.1 Definition of Enterprise Model and Corporate Data Model

The definition of an enterprise model in the proposed ISO 14258 standard *Industrial Automation Systems - Concepts and Rules for Enterprise Models* [WINC99] is:

> "a representation of what an enterprise intends to accomplish and how it operates, which is used to improve the effectiveness and efficiency of the enterprise."

The following note elaborates on the definition:

> "An enterprise model is an abstraction that identifies and represents the basic elements of an enterprise and their decomposition to any necessary degree that is used to improve the effectiveness and efficiency of the enterprise. It also specifies the information requirements of these elements, and provides the information needed to define the requirements for integrated information systems."

Since the ISO standard is concerned mainly with automated systems, the enterprise models referred to by the standard setters are not necessarily representative. An enterprise model may have a different purpose, such as documentation, internal standard setting, data warehousing, information architecture planning and the like. Hence a more inclusive definition is preferred.

[FOX93b] defines an enterprise model as follows:

> "a computational representation of the structure, activities, processes, information, resources, people, behaviour, goals and constraints of a business, government, or other enterprise."

The model can be either descriptive - describing an existing situation - or prescriptive - describing a desirable goal. This definition makes it clear that both static and dynamic aspects need to be captured in the model. A typical enterprise model will therefore include a number of views.

A subset of the enterprise model is the corporate data model:

> "an abstract representation of the information requirements of all or part of an organization, independent of functional boundaries with an organization and implementation technology" [BRAN86].

The focus of the corporate data model is on the static information aspects, leaving out dynamic aspects of the enterprise such as processes, activities and behaviour. Some authors refer to the corporate data model as the organizational or enterprise data model. The literature is divided on the exact differences between the concepts of enterprise, corporation and organization although it appears it is mainly a matter of emphasis. The following descriptions are an attempt to summarise common usage as reflected in the literature and a number of dictionaries. It must be stressed that no description is meant to be definitive or even universally recognised (see the discussion of "concept gradience" above). Also, most of the terms have multiple meanings:

- **Business**: a private organization aimed at making a profit from commercial activities. A close synonym is "**firm**". Can be very small in size as in a one-man business or small partnership.
- **Enterprise**: an undertaking or business activity. The emphasis is on the activity, not necessarily formalised and not necessarily but most often for profit. A close synonym is "**venture**". Can be used where the life is fairly short as in "virtual enterprises" (in the meaning of different businesses partners working together for one specific project).
- **Corporation**: a group of people formally acting as a single individual, especially in business. This stresses the independent (legal or formal) status and external identity. In many contexts, corporation refers to a specific legal statute in terms of company legislation. A close synonym is "**company**". Often carries a connotation of "largeness" or "big size".
- **Organization**: a structured body of people, a controlled and planned system. The emphasis is on the structure and relationships of the group, usually closely tied to the overall goal or purpose for which the group is organised. Often used when there is no legal status in local company law; also used for the many non-profit organizations. The notion of an organization encompasses all of the previous concepts and more, for example a church or a sports club, except that it typically requires a group i.e. a number of persons. It is therefore not typically used for small or one-person businesses.

In what follows, the distinctions - if any - between organizational, corporate and enterprise models will be ignored although a business model is perhaps more restrictive in that it may have a specific implied goal to generate profit.

[WORT00] suggests the following historical evolution in enterprise modelling:

- Late sixties and early seventies: the development of modelling frameworks, especially defining modelling constructs and notations e.g. structured analysis and design techniques or IDEF.
- Seventies and early eighties: the development of (mainly academic) methodologies and project management environments for enterprise modelling such as GRAI or GERAM (see enterprise integration).
- Eighties and nineties: impact of enterprise modelling on commercial, practical methodologies and tools e.g. those used in the development of Enterprise Resource Planning systems: ARIS (for SAP R/3) and DEM (for Baan).

2.8.2 The Level of Detail in, and Size of, an Enterprise Model

An enterprise model or a corporate data model does not require the same amount of detail as a model intended to serve as the basis for application development. In a corporate data model, For example, many detail attributes, subtype entities and domains could be omitted. Attempts to include too much detail doomed many enterprise modelling efforts and hampered the understanding and verification of the ones that were completed [SMIT98].

Indeed, the problem of representing large models is seen as one of the more serious challenges facing the modelling community. This is especially the case for the "flat models" still employed widely in the *data* modelling community where it has become known as the *database comprehension problem* [CAMP96]. Newer modelling approaches try to alleviate this problem by providing grouping constructs, such as the UML "package", and modelling tools commonly have multi-level model support by means of zoom in/out and expansion facilities.

Thus the use of different levels when building an enterprise model is advocated. [INMO99] suggest the following three levels in the context of a corporate data model:

- The high-level model containing very broad definitions and all major categories i.e. the major entities, their definitions and the relationships between them.
- The mid-level model includes the details of each subject area, including keys, and attributes.
- The low-level data model contains the information about specific data design, possibly including physical characteristics such as data types and indexes.

The debate about whether to include the low level in an enterprise model will depend on the purpose and context of the model. When defining an enterprise architecture, the first two levels will suffice; whereas the building of an ERP or data warehouse (which forms the background to Inmon's approach) will require a fully-fledged low-level model. In many models, additional levels are used so as to limit the number of concepts in any one sub-model approximately to the magic 7 plus or minus 2 entities - supposedly representing a human cognitive short term capacity limit [WITT94]. It is generally accepted that design or implementation-specific concerns such as keys, data types, interface objects and the like are not normally necessary in the enterprise model [SMIT98; BECK00] although a special exception is often made for relationship entities resulting from data normalization.

Another guideline about the inclusion of keys and attributes in the mid-level model is based on the following taxonomy of entity attributes (with examples) [NEWT96]:

- Identifying: name, context, identifier, version, registration authority, synonyms.
- Definitional: define, explain and/or illustrate the concept.
- Relational: classification scheme, keywords, related data, type of relationship.
- Representational: representation category, form of representation, data type of data element values, range and domain of data element values, layout of representation.
- Administrative: responsible organization, registration status, submitting organization, comment.

Based on this, it is proposed that a high-level model should include only the definitional and relational attributes, along with those identifying attributes that are of a semantic nature (i.e. name, context, synonyms).

Because of their scope, enterprise models tend to be huge and rather unwieldy. A typical example is the corporate data model of Saskatchewan Wheat Pool, a large North-American agro-business, consisted of about 500 entities (conflated into 300 tables) with about 10,000 objects. The entire

repository comprises about 6 megabytes [LOCH98]. As will be seen, this is fairly representative of a typical enterprise model.

2.8.3 The Purpose and Use of an Enterprise Model

Many enterprise models have been developed with a *single specific purpose* in mind e.g. BPR (Business Process Reengineering), ISO 9000 certification, a data warehousing or systems analysis project. These "throw-away" enterprise models [WHIT97a] tend to lose accuracy and relevancy quite quickly, due to the fast-changing nature of the enterprise and its environment.

ISO 14258 states the *purpose* of enterprise models as follows:

> "[enterprise] models can be constructed to analyse, guide the engineering of, and manage the operation of enterprises." [WINC99]

Note that this was in the context of enterprise engineering. More generally, [KIRI00] states the uses of enterprise models as follows:

- **Business analysis**: problem detection.
- **Business process re-engineering**: defining new processes or developing new systems.
- **Requirements engineering**: facilitates the definition of the requirements specification.
- **Organizational learning and knowledge management**: forms the basis of knowledge propagation and amplification.

Whitman ascribes some additional uses to enterprise models [WHIT97a]. An enterprise model:

- Facilitates **communication** by providing a common language and understanding of processes.
- Serves as a baseline for the continuous **improvement of existing processes** (BPI).
- **Documents** existing processes and structures e.g. for new staff induction, ISO 9000 certification.
- Facilitates the **control** of real world business (management, optimization, simulation, what-if scenario building).
- Can be used to **integrate** the **information** resources, systems and procedures of merging organizations.
- Is a starting point for **information systems development** including operational systems, database design, data warehouses, decision support system (DSS) and executive information systems (EIS).

Corporate data models, more specifically, are claimed to result in the following **benefits** [LOCH98; VANS00]:

- Better data architecture and database structure.
- Less data duplication/redundancy and fewer system interfaces leading to lower system coupling.
- Lower system development cost.
- Faster processing speed due to less data fragmentation and higher system integration.
- Better quality data due to standardized data formats and reduced data redundancy, allowing better data update control.
- Reduced technology mix and resultant smaller spread in skills requirements.

2.8.4 How Do Enterprise Models Differ From Other Domain Models?

The most obvious is the domain scope of the enterprise model: the entire organization. Since the domain is large, dynamic and complex, the resultant model is equally large and complex. This is unlike many engineering and computer science models of artefacts which are more controllable and exhibit predictable behaviour and more stable internal structures: although the solar system (as seen through an astronomer's eye) is physically much larger, it is arguably a far less complex system than your local corner shop since the former can be modelled in a few sophisticated but deterministic mathematical equations. Some of the issues of size and complexity have already been dealt with in the section on model views and levels.

The dynamic aspect - the continually changing domain - requires that the enterprise model must be continually updated to remain representative. The term *living enterprise model* has been suggested to describe the type of active model that remains current, auditable, and hence valid. The following are the main characteristics of a living enterprise model: maintainable, dynamic, expandable, de-compositional (multiple levels), consistent with enterprise metrics, driven directly from actual enterprise data, capable of simulation and allowing non-standard activities to be accommodated [WHIT97a]. Unfortunately, the literature is divided on how to build this living model.

Another peculiar aspect of the enterprise model is that it can become part of the system it is modelling, either indirectly as an information repository which forms part of the information resources and flows in the organization, or much more directly where the model participates in the organizational processes. This latter type of model is referred to as model enactment or executable model. Automated factories, supported by sophisticated CIM systems, are the textbook example. In fact, the entire discipline of "enterprise integration" (as discussed in the next chapter) is devoted to developing the necessary frameworks and constructs to drive this development. The self-referential aspect of executable or self-enacting models introduces unique modelling challenges (see e.g. [BRUN97]). Halfway between the descriptive and executable models are the "intelligent" models: models, often in formal notation, which possess logical ability and allow shallow or deep reasoning. For instance, given that an organization's structures, roles, goals and resources are modelled, a specific person's resource allocation could be deduced automatically given his or her role in the organization [FOX98]. Many of these models feature in the ontology discipline.

A final key difference between "ordinary" and enterprise models concerns the introduction of a number of abstract, organization-specific modelling constructs. Although each modelling method has its own set of modelling constructs, as discussed in the sections on meta-modelling and semantic relativism below, many authors propose an additional set of more specific modelling constructs to aid enterprise modelling specifically.

An example is the set of concepts introduced by Marshall in his high-level organization model [MARS98; MARS00] as illustrated in Figure 2-2 below. Although he uses three fundamental types of business objects in his model implementation (purpose, process and entity), his meta-model is based on three types (role, state and activity), which are orthogonal to three aspects: rule, plan and act.

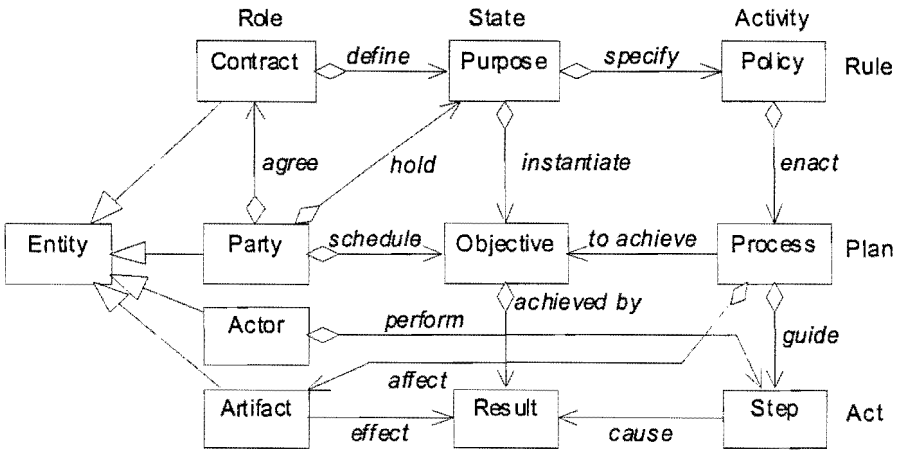


Figure 2-2 Marshall's High-level Organizational Model [MARS00]

Other authors similarly introduce their own sets of enterprise specific high-level concepts, such as the Inspired Frameworks Process Architecture [MCLE01]. Many others will be introduced in the main research discussion. Although these constructs, as abstract object types, can be seen as the upper-end of the enterprise model, an alternative is to see them as a kind of meta-model. Perhaps the best way to describe them is with the term "*mesa*-model" i.e. at a domain-specific intermediate level between model and meta-model.

2.8.5 Some Typical Problems with Enterprise Models

The following problems are typically encountered with enterprise models; though perhaps not specific to enterprise models, these problems are often more pronounced in the context of the model size and scope [HUBE94; BROD82].

- The development of the model is not documented, in particular the reasons or motivations for particular classifications and generalizations.
- After the initial information gathering phase, concepts are consolidated, standardized, abstracted and generalized, necessitating the use of new terms and concepts not readily understood or accepted by the user departments that were initially involved. Often the cross-referencing of the new "unified" terminology to the user terms is not documented explicitly in the model. This problem is exacerbated in larger organizations where the same term is given different meanings by different departments (e.g. asset, cost centre, order etc.)
- An "ideal" model is developed but takes the point of view of the IT department. The focus is on optimizing processes or the automated information processing potential. This is linked with the "missionary" bias of the IT staff where 1 or 2 percent of the organization's staff tends to "dictate" the terminology use for the rest.
- The excellent graphical capabilities of today's sophisticated tools make it easy to create copious volumes of nicely formatted documents. This great volume overwhelms users and the quality of the document presentation may disguise bad modelling or weak methodologies.
- There are still some gaps in modelling theory, which manifest themselves when modelling complex real-world domains such as enterprises. In particular, there are still unresolved issues in respect of modelling dependencies, incomplete knowledge, exceptions, anomalous information etc.

2.8.6 Reference Models, Generic Models, Templates and Frameworks

The purpose of a *reference model* is to serve as a guide for developing specific systems. Its main purpose is the stimulation of ideas and to provide a structure with pre-built components which can be used as is, changed or omitted. Often reference models represent generalizations or sub-sets of concepts [BAUE98]. Whereas an enterprise model refers to a specific organization, a reference enterprise model models the "*typical*" organization".

For purposes of this research, no distinction will be made between *reference* and *generic* models. There appears to be an academic distinction in that reference models have a connotation of authority or standardization and hence a more prescriptive flavour. With a generic model, the emphasis appears to be on the commonality between specific models or the sub-set of universal model elements. From the literature survey, it appears that this semantic distinction is all but ignored by practitioners or in the model content.

Reference models differ from model *templates* in that a template is usually derived from previous modelling experiences and therefore at a fairly detailed level, often focussing on one particular area or situation. Reference *frameworks*, on the other hand, are at a higher level of abstraction. A framework is mostly concerned with the modelling environment, approach and process, and provides guidance on how to model in specific circumstances. A reference modelling framework may be concerned with how many and which views to adopt, what meta-model to use and how to go about the modelling process. This is in contrast to a reference model which is typically coded using a particular model notation and often restricted in applicability to a fairly specific modelling domain.

2.9 Modelling Languages and Knowledge Representation

The issue of model representation - the third vertex of the meaning triangle in fig 2.1 - is an extremely important one: what language is used to express the model and what are the allowable modelling constructs. The *representation* of an enterprise model in a specific language has to do with the model syntax – as opposed to the model semantics which denotes the *meaning* of the model. ISO14258 defines the model syntax as referring

> "to the permissible arrangements of the representations of the elements and to the permissible kinds of relations".

The representation language of a model fulfils a number of different roles or purposes [DAVI93]:

- It is a **surrogate**: we can reason with or manipulate the model instead of having to take action within the real world enterprise.
- It is a set of **ontological commitments**: a decision is made on what can be seen in the real world (and how it is seen) i.e. like "a strong pair of glasses that determine what we can see, bringing some part of the world into sharp focus, at the expense of blurring other parts." [DAVI93:4] This has the disadvantage that certain aspects are left out (unavoidable, since abstraction necessitates it) but also the advantage of reducing the real world's complexity by allowing us to focus on the important or relevant aspects of real world.
- It is the medium for **model computation** (e.g. automatic deduction or code generation). Generally there is a trade-off between the computational efficiency (an important pragmatic consideration) and the expressive power of a modelling language.
- It serves the purpose of **communicating** the model to or between **humans**. The ease of interpretation is an important issue since model validation is primarily done by humans.

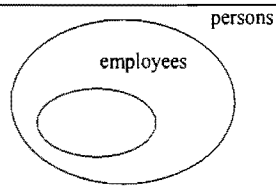
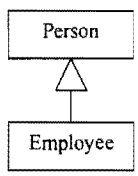
These roles will form a thread through the remainder of this discussion.

2.9.1 Modelling Language Taxonomy

A model could be represented using the UML, first-order logic (FOL) or English. Each of these languages prescribes a particular vocabulary as well as rules of grammar that lay down how to construct "well-formed sentences" i.e. legal combinations of vocabulary words. English has a very large vocabulary (and most words have more than one meaning) and a very flexible grammar, whereas FOL uses a very restricted set of "key-words" and mathematical symbols with a very precise grammar. The UML uses a variety of graphical symbols as its "vocabulary", e.g. a rectangle to denote a class or object instance, which can be combined only in a specified way (the "grammar" rules) For example, a generalization (subtype/supertype) relationship arrow is a connector between two classes or two relationships, but can not connect two comments or a relationship with a class. UML also specifies the *meaning* of the generalization relationship i.e. in terms of inheritance of attributes.

As an illustration of the possible variety, Table 2-1 models the same model fragment – the fact that employees are persons –using a number of commonly used modelling languages.

Table 2-1: Different representations of "Employees are persons"

English	"An employee is a person." (statement) "If an entity is an employee, it must be a person" (rule)
Dutch	Een personeelslid is een persoon.
XML (Extensible Mark-up Language)	**<complexType name**="person"> **</complexType>** **<complexType** **name**="employee" **base**="person" **derivedBy**="extension"> **</complexType>**
First-order logic	(∀ x) (employee (x) ⇒ person (x))
KIF (Knowledge Interchange Format)	(forall (?x) (=> (member ?x employees) (member ?x persons)))
JAVA	public class Person { … } public class Employee extends Person { … }
Set theory (Venn diagram)	
UML	

Myers proposed three well-know taxonomies of programming and specification techniques: one for programming systems, one about program visualization and one for specification techniques. The latter is of particular interest to modelling and [MYER90] lists the following techniques:

- Textual languages
- Flowcharts
- Flowchart derivatives
- Petri nets
- Data flow graphs
- Directed graphs
- Graph derivatives
- Matrices
- Jigsaw puzzle pieces
- Forms
- Iconic sentences
- Spreadsheets
- Demonstrational

The following discusses some important aspects with respect to the options and choice of a suitable modelling representation.

2.9.2 Graphical Versus Textual Representation

A continuing debate in the area of modelling is whether modelling languages should take a graphical form or a linear/text form [SARR96].

For the interface with humans, the so-called *external* representation of a model, graphical languages seem to have a strong intuitive appeal and, although not everyone is equally adept at interpreting two-dimensional graphical representations, there is evidence that this skill is at least partially an acquired one [KREM98]. There are a number of important psychological advantages in favour of using visual languages:

- Human abstract reasoning appears to be pictorial in nature; and some authors claim that most thought processes may be transformations of pictures. In any case, pictures appear to be computationally much more efficient for human computation than linear text.
- Visual organization of data is a way to increase the limit of the capacity of short term memory by using the chunking mechanism.
- The human mind can store sensory data in great detail.

It must be noted, however, that these claims are not necessarily shared by all researchers. For example, [JARV88] claims that "most of the proponents of graphics have never tested their claims. Further, when those tests are performed, the results are contradictory and inconclusive." Some tests have indeed indicated that especially novices may have more difficulty in interpreting visual data than textual expressions. [GREE91] reports that a number of tasks took longer when subjects were given graphical instead of textual expressions. More recent research is less critical, although [KELL98] still points out the lack of empirical evidence. Perhaps the evolution of computer technology (faster hardware with more sophisticated GUIs) has swung the balance conclusively in favour of graphics?

Graphical model representation remains thus the most popular mechanism for documenting and specifying models to humans. In fact, considerable research is underway to investigate the use of three-dimensional (3D) modelling notations. [KENT97] argues that one single 3D model can integrate all the information of several types of two-dimensional diagrams: "indeed, it appears that the 2D UML diagrams are simply projections of the 3D model".

2.9.3 Formality of Model Specification

In contrast to the "human interface", computer storage and processing of any model is normally facilitated by means of an appropriate *internal* representation language which is non-graphical in nature. Computer science has the entire sub-discipline of *Knowledge Representation* (KR) devoted to the study of the philosophical foundations, expressive power, computational efficiency and design issues of languages for the computer processing of knowledge. Sarris puts it as follows:

> "from ways to conceptually model enterprise semantics in the form of objects, processes, relations and rules, languages and fundamental constructs for integrating heterogeneous models, to CASE and data dictionary/repository technologies for effective model management and reuse, each challenge emphasizes a different aspect of the same underlying problem: knowledge representation" [SARR96].

A more recent, quite comprehensive overview of KR can be found in [SOWA00]. It must be noted that, for our purposes, the important issue is not the technical one of *internal* memory representation of knowledge (i.e. the binary code) but the "apparent" internal representation used to ultimately mediate the model to the user (normally through an additional "top layer" front-end user-interface, which is often GUI-based) or used to interchange data between modelling tools.

The main concern with a computer-centred modelling language is its degree of formalism. The following distinctions can be made, although in reality these form part of a continuum [LUKO96]:

- **Formal** techniques have a formal syntax and semantics. They are typically based on a mathematical theory such as set theory or logic algebra. Examples are KIF, PIF, KL-ONE, etc.
- **Semi-formal** notation has a formal syntax but informal semantics. Examples are Conceptual Modelling Language (CML) or UML (without OCL annotation; [GEIS98a] argues that it can be considered as a formal language if OCL is added, see also [EHRI99]).
- **Informal** notation has an intuitive syntax and semantics. Examples are natural language (English), Object Role Modelling (ORM) [BECK98] and mind maps.

In theory, a formal technique will, given the appropriate deduction engine, not only allow for fully automated deduction abilities - including automatic code generation and database interrogation, but it can also serve as the foundation of a fully executable model [EHRI99]. In fact, the development of a *model-equivalent* computer programming language has been argued i.e. where the same language is used to analyse as well as implement (or code) the system. [LIDD95] discusses some of the theoretical difficulties including Turing completeness and impedance mismatches, e.g. conflicts between variables, type and classes; persistent versus transient data; imperative/declarative processing conflicts etc.

The trade-off for the power and precision of a formal language is the fact that formal model specification appears to be a tedious, error-prone and labour-intensive process, judging from the many person-years of labour that have gone into the development in enterprise ontologies, for example. Arguments in favour of formal methods notwithstanding (see the "seven myths of formal methods" [HALL90] and "seven more myths of formal methods" [BOWE95]), formal specification techniques have not made major inroads yet into the enterprise modelling world.

[DAVI93] argues that the selection of the KR language (and its degree of formality) should match the "spirit" of the language with the domain at hand, rather than the forced use of one specific "use-for-everything" language which necessitates the invention of all sorts of creative work-arounds, extensions and ad-hoc constructs. Since enterprise models are often meant to bridge between an

informally stated purpose (the requirements) and a formal system implementation, it is perfectly acceptable for a model to be specified using a semi-formal notation.

2.9.4 Expressiveness and Semantic Relativism of Modelling Language

The representation ability of a modelling language is composed of two elements [SALT93]:

- Its **semantic relativism**: the degree to which it can represent different conceptions of the same world.
- Its **expressiveness**: the degree to which it can directly model any particular real world concept.

The semantic relativism of a graphical language is directly related to the number of different views supported by the modelling language, e.g. the UML has 9 or 10 different diagrams, each modelling a particular model view whereas an ERD or DFD each represent only one specific view. For modelling enterprises, it is necessary to be able to model dynamic as well as static aspects; an ERD or class diagram does not allow for the dynamic aspects. General KR languages with a few very primitive constructs, e.g. node, link, slot, and grouping, are much more relativistic from a semantic viewpoint.

The expressiveness of a language relates to specific language constructs (or primitives) that support the modelling of a particular domain. For instance, if one models an enterprise, it helps if the modelling language provides enterprise-specific constructs. Table 2-2 below lists some typical primitives (and their variants) against which [MORA94] evaluated a number of KR languages. Including more primitives into a KR language increases its expressive power and makes for much more succinct models. [GOGI95] specifically compares and evaluates the expressiveness of KR symbolisms in terms of their "representational succinctness" i.e. how few constructs they need to represent a model. The trade-off lies in the computational overhead as well as the increased complexity of the modelling language. Also, an increase in primitives often allows multiple alternative ways of modelling a given situation, an attribute which makes model translation, inspection and validation much more difficult.

Table 2-2: Some enterprise modelling primitives in KR languages [MORA94].

Primitive	Variants of primitive
Activity	Plans; business processes; events
Goals	Task; options; goal; business concern; requirements; objectives
Agent relationship	Responsibilities; delegation; authority; agreements; commitments
Resource	Agents; place; space
Time	
Organization	Culture; policy; groupings
Uncertainty	

With respect to the modelling language expressiveness and semantic relativism, there is a choice between the following modelling paradigms:

- A combination of **classic systems engineering modelling techniques**. Classic SE techniques such as DFD, ERD, etc represent one particular modelling view and use a number of generic, i.e. non-enterprise specific, modelling constructs. Many traditional methodologies combine several techniques to provide multiple perspectives. This necessitates specific verification procedures to ensure consistency between the model views.

- An **object-oriented approach** of which UML is perhaps the most representative and popular example. Although it allows many different views, its expressiveness with respect to the enterprise domain is rather low. To increase the expressiveness, many enterprise modellers have extended UML with enterprise-specific constructs such as contracts, resources, goals and activities. Refer to the "mesa-models" by e.g. [MCLE01; MARS00].
- A **knowledge representation approach** which allows for a higher degree of expressiveness. Many KR languages have been designed specifically for modelling within an organizational context and explicitly incorporate a number of the modelling primitives listed in the table above. An example is MEMO (Multi Perspective Enterprise Modelling) [FRAN99a].
- **Natural language**. This is the most expressive knowledge representation language but generally not considered formal (specific) enough for modelling purposes.

Whilst it may initially appear that the most expressive languages are preferable, the issue is not quite as simple. It is generally accepted that the new OO paradigm is much more natural and powerful than earlier RDBMS-oriented techniques. There is often a fairly easy migration e.g. from an ERD to an Object Class diagram. [CHU97] points out that OO allows a much more natural modelling approach and that it is easy to add additional complexity to the model. [SALT93] also discusses how OO models are more expressive than relational and extended ER models. [YEO96] argues that OO is not only a better paradigm for the "external" representation of enterprise models, but also motivates that the internal representation, i.e. within the modelling tool, should be OO-based.

However, [HAY00] points out that the benefits are not automatic. In defence of his generic (ER-grounded) data models, he points out that many object modellers do not necessarily apply the necessary discipline to make use of the specific benefits of OO. For example, "where an E/R model would represent the employment of a person by a company, an object model might simply show an employee". This criticism, however, is more in respect of the (incorrect) use of OO rather than the intrinsic expressiveness of the OO paradigm. One remaining theoretical problem is that of closure: in the relational data model, applying relational operations to relations always produces relations; hence its conceptual model is mathematically closed [BROD82]. This is not true for the object-oriented model whose logical foundations are still not fully worked out. Finally, there tends to be a higher number of object types in OO than the equivalent ER data model, mainly due to the creation of abstract super types. However, these criticisms have not stopped the OO paradigm from making steady but certain inroads in the enterprise modelling world.

The real debate at the moment is whether to adopt a KR or an OO approach; and here the issue is far from settled. By far the dominant paradigm used by IS practitioners is the OO model, typically by means of UML diagrams. However, it is felt by many researchers that OO is not powerful enough to represent models. The following argument is made in defence of CML (Conceptual Modelling Language):

> "We consider that pure rule representation as well as an object-modeling language, data dictionaries, entity-relationship diagrams, among other methods are considered no longer sufficient neither for the purpose of system construction nor for that of knowledge representation. We believe that knowledge is too rich to be represented with the above-mentioned methods. This requires stronger modelling facilities." [CAIR98]

Two specific arguments are mentioned by [HULL90]: OO models do not have the rich type constructors of, for example, semantic models (e.g. the grouping constructor which builds a finite power-set of existing elements of an existing subtype) and the different types of inheritance (e.g. in

semantic models, inheritance is behavioural, not structural, so that seemingly unlike types can inherit methods). [PARP98] compares conceptual graphs (CGs) against OO networks and states that, since CGs map directly to FOL, it is much easier to prove the correctness of CG models.

Most of the criticism is levelled, not necessarily against OO, but more specifically against UML. It appears indeed that a substantial amount of "UML" conference papers (see e.g. the OOPSLA conference series) deal with how to extend UML for specific modelling situations. Despite its many different diagram types, there is still a strong feeling that some model views are missing, such as enterprise process models [MCLE01]. UML models can be extended to some extent e.g. by means of the stereotype construct, as suggested by [MARS98; BERG00; HALP01].

On the other hand, modelling in a more expressive language requires more computational overhead, poses the modeller and model user/validator with significant cognitive complexity, and introduces redundancy by providing many alternatives to model the same situation. As mentioned before, it may not be necessary to require such a high degree of formalism or to capture the domain in all its complexity [MENZ98]. In fact, in the particular case of ontology modelling languages, [CRAN99] discusses a number of commonly used KR languages such as KIF and KL-ONE and argues that a combination of UML and OCL (Object Constraint Language) has the same richness as any other KR system but without their steep learning curve and "relative obscurity outside the AI laboratories".

A final observation is that the data/meta-data distinction becomes more and more blurred as one moves from the relational model (which has an extremely clear separation between the data structure and the data contents - the so-called *intensional* and the *extensional* information) to the OO model and then to the KR languages which often store meta-data and data together using the same constructs.

It is the author's view that the model-driven approach will, in time, force a shift from the OO modelling approach to the even more expressive KR modelling paradigm.

2.9.5 Some Typical Enterprise Modelling Languages

A large number of representation languages are used for enterprise modelling. As part of the PSL (Process Specification Language) standardization effort, a detailed inventory of the following twenty-six *process* representations was made [KNUT98]:

- ACT: a domain-independent formalism to specify knowledge about activities to generate and/or execute plans
- A Language for Process Specification (ALPS): aimed at discrete-process manufacturing.
- AP213: the application protocol for sharing STEP (Standard for the Exchange of Product model data - an ISO 10303:1992 standard) data.
- Behaviour diagrams: describe system functionality for systems engineering (SE) software.
- Core Plan Representation (CPR) for planning systems support.
- Entity-Relationship Diagram (ERD) models information primarily for relational database model implementation.
- Functional Flow Block Diagrams (FFBD) charts system functionality and sequencing.
- Gantt Charts graph projects and schedule resources and activities.
- Generalized Activity Network (GAN): a PERT-like but more general chart representing activities as links between states.
- Hierarchical Task Networks (HTN): used for AI planning systems.
- IDEF0 is a SADT inspired standard for functional modelling.
- IDEF3 describes the behavioural aspects of as system using process flows and object states.
- <I-N-OVA> (Issues, Nodes, Ordering, Variable and Auxiliary constraints) models tasks, plans, processes etc. as constraints on system behaviour.

- Knowledge Interchange Format (KIF) aims at sharing knowledge using a standard FOL-like but ASCII-based syntax. Has a linear and tree-like version.
- Program Evaluation and Review Technique (PERT): a network analysis technique for projects with uncertainty.
- Petri Nets: a graphical language for modelling nets with concurrent, interdependent events. Has many variants for specific applications.
- Process Interchange Format (PIF) supports the exchange of enterprise process models between systems.
- The others mentioned by Knut are: O-Plan, OZONE, PAR-2, PART49, PFR, Quirk, VPML. He also includes a number of "supporting representations", namely the AND/OR graphs, Data Flow Diagrams (DFDs), Directed Graphs, State Transition Diagrams and Tree Structures.

A large number of other KR languages exist, many not geared specifically to process modelling. Not all have necessarily been used in an organizational context. [LUKO96] and [MORA94] also evaluate ML, KARL, KADS and CommonKADS, LOOM, OMOS, NIAM, INFO-MOD, Model-K for enterprise modelling suitability.

A more recent paper [VERN97] reviews the following commonly used languages for enterprise modelling, and compares them on the basis of a list of essential requirements for enterprise modelling:

- The CIMOSA language: part of a reference framework for enterprise modelling and integration.
- ARIS ToolSet language: developed as part of a comprehensive enterprise modelling tool set which served as the basis for developing SAP R/3.
- ER models / EXPRESS: EXPRESS is a more formal data description language, based on the ERD model, with particular support for STEP entities.
- GRAI nets: static descriptive models for enterprise integration.
- IDEF suite of models - produced by the ICAM project. In addition to IDEF0 and IDEF3, IDEF1x is a type of ERD and IDEF2 provides a graphic, dynamic model based on the SLAM simulation language.
- IEM: another reference architecture for enterprise modelling which mostly covers the function and information views using an OO perspective.
- OOA / OMT: Object-oriented graphical notations (now superseded by the UML).
- Petri nets: see above.
- SA/RT: a semi-formal analysis and design technique dedicated to real-time and reactive systems

More recently, unification efforts in the software engineering field have led to the increasing acceptance of UML as a graphical tool for enterprise modelling [BRUNO97; MARS00]. UML consists of a number of formalisms, some of which are specifically geared towards models for software development, whilst others are generic modelling tools.

As a *lingua franca* for the *exchange* of models between *modelling tools*, it appears that there is a move away from CDIF (CASE Data Interchange Format) [LEME98] to the XML-based standard XMI (XML Metadata Interchange) [COVE01]. The *de facto* standard for exchange of data between *KR systems* is still KIF although XML-based proposals have been floated, notably OIL (Ontology Inference Layer; and its derivatives) and XOL (XML-based Ontology exchange Language) [DIMI00].

2.10 Modelling Tools

An important pragmatic aspect of enterprise modelling concerns the selection of the appropriate modelling tool. Since many of the modelling tools are used in the context of information system development, the rapid changes in the technology environment are reflected in a very dynamic and volatile tool market. The intention of this section is therefore not to discuss individual tools but rather to give a quick overview of the concerns related to tool selection viz. a broad categorization of enterprise modelling tools and the criteria one should consider prior to selecting an appropriate modelling tool.

2.10.1 Tool Classification

There used to be fairly precise distinctions between the various classes of tools. Now, however, technological developments have blurred these boundaries to such an extent that the following list has become more like a set of reference points on a modelling tool continuum.

- **Graphical Modelling Tools.** Many pure "model diagram drawing tools" were developed during the 1980s. Since these tools require an internal diagram repository anyway, the surviving tools have extended their capabilities to include at least superficial code (e.g. "stub" generation i.e. class and attribute definitions only) or database schema generation, and more substantial documentation abilities. For example http://www.methods-tools.com/tools/modeling.html lists 74 (mostly UML) graphical modelling tools (many also IDE & CASE) of which only four remained pure graphical tools: GOOFEE diagrammer, MagicDraw UML, Playground and VisualThought. All others tools incorporate some degree of code generation. In the KR field, there are still a number of graphics-only tools for generation of conceptual graphs or Petri-nets, for example.
- **CASE tools.** Computer-Aided Software Engineering tools are specifically designed to support a model-driven system development process. Some CASE tools are therefore integrally tied to specific methodologies, though many of today's tools support multiple methodologies or allow methodology customization. A distinction used to be made between [VANB99b]:
    - Upper-CASE (U-CASE): focussing on the analysis stage i.e. requirements modelling and prototyping;
    - Lower-CASE (L-CASE): concentrating on the design stage i.e. code generation, testing and implementation (i.e. not useful for enterprise modelling); and
    - Integrated-CASE (I-CASE): supporting the entire SDLC, usually adding comprehensive project management tools.

  This distinction is now also rapidly becoming obsolete since most contemporary CASE tools support most stages of the SDLC, though not all tools feature full project management support. A recent list (available from http://www.objectsbydesign.com/tools) mentions 601 CASE tools (as of 20 July 2001) although some entries reflect different tools from a single vendor I-CASE tool suite. The competitiveness between these tools is quite fierce: http://www.cetus-links.org/oo_ooa_ood_tools.html lists 58 downloadable tools - some full, others limited versions - and some market shake-out is to be expected. Some of the higher-end tools (as opposed to the CASE "toys") are Cayenne, IBM UML Designer, Sapiens and Stirling's suite of COOLGen/Spec/Jex [MCLE01].
- **Integrated Development Environments (IDEs).** Initially used for stand-alone programming GUI applications (akin to single user L-CASE tools), IDEs have grown into multi-user environments with sophisticated database capabilities and (often UML) diagramming tools, thus blurring the distinction between the high-end IDEs and CASE tools. Examples are Jade, PowerBuilder, Delphi.
- **Meta-modelling tools.** These tools are not modelling tools per se but allow one to define or specify one's own modelling elements and diagrams thus generating customized (integrated or stand-alone) modelling tools. Examples are Meta-Edit/Meta-CASE, MethodMaker (Mark-V) and IPSYS tool-builder [MART96].
- **KR tools.** The KR discipline concerned with the development and specification of specific KR languages developed its own set of tools for each of the KR languages. Examples are METIS,

CommonKADS, LOOM editor and many more with esoteric names such as GRAIL, KRIS, CRACK, RACE etc. One might claim (not quite wholly tongue-in-cheek) that there is one KR tool for each KR researcher or doctoral student. The tools most relevant to enterprise modelling are the ontology editors such as OntoEdit, Kactus, OntoBroker, ODE (Ontology Design Environment). [BRAZ98] developed a framework for the evaluation of the suitability of KR tools for knowledge modelling and compared the following tools: Desire, CommonKADS, Protégé, Mike, Vital and Task.

- **Knowledge Management Repositories.** KM repositories are extremely flexible databases geared towards the customsd storage and management of diverse knowledge. These interesting tools can and have been used for corporate model management. The following examples will give a feel for the differences in approach and implementation: ConceptBase ("a deductive object manager for meta databases"; http://www-i5.informatik.rwth-aachen.de/CBdoc/); Principia ("a configurable data repository usable for corporate data modelling"; http://www.principia.co.uk); k42 ("a knowledge structuring tool for semantically connecting data is disparate heterogeneous data sources by means of Topic Maps"; http://k42.empolis.co.uk) and Archi ("a web-based knowledge repository suitable for enterprise architecture management"; http://www.inspired.org/html/archi.htm).

In addition, there are a number of miscellaneous tools such as external metrics collectors (though usually taking code as input e.g. Resource Standard Metrics; http://www.m2tech.net and McCabe IQ2; http://www.mccabe.com), validators (to set and enforce syntactic and semantic naming standards in models; e.g. Kismeta [ORLI99]) or IKARUS/ClearTalk (a knowledge management system with a semi-controlled structured English natural language front-end; http://www.site.uottawa.ca/~kavanagh/Ikarus/IkarusInfo.html).

2.10.2 Criteria for Selecting Tools

There are many criteria which could be considered for evaluating a modelling tool. The following criteria were suggested by the author in a student assignment to evaluation UML modelling tools: price, availability, local & internet support, market share (difficult), availability of a demo version, which UML (and non-UML) diagrams are supported, how pure OO is it, maturity of the tool, scalability, class libraries provided, open/proprietary and Corba/DCom compatibility, checking & testing support, code generation (and, if so, which languages), user-interface, ease of use & learning curve, integration with other tools, running platform (and required hardware specifications), delivery platform, methodology, project management & group support, documentation.

In the context of enterprise modelling, the following additional criteria could be added: degree of model verification, specific business process re-engineering functionality, model simulation capability, architecture modelling support, collaborative web development, pattern support, reference models provided, formal language support (KIF?), full model life-cycle management, meta-modelling capability, model metrics provided, HTML & MS-Office documentation, intranet support, prototyping/RAD (Rapid Application Development), reverse engineering, repository technology, CDIF and XMI export/import.

2.11 Meta Modelling

This section introduces the concept of meta-modelling. Due to its importance to the research, it is dealt with in some more detail.

2.11.1 What is Meta-modelling?

Any modelling process, by necessity, implies another modelling system at a higher level [LYYT99]. This higher system, one level up, is called the meta-model, and specifies how the modelling process must happen and/or what the model can look like.

Meta is a common Greek-derived prefix that means "after" and refers to something of a higher, second-order kind. It typically entails a recursive or self-referential relationship. Meta-data (or metadata) is data that describes other data (data about data). A Meta-language is a language used to describe other languages. A metafile is a file that contains other files. The HTML META tag is used to describe the contents of the web page in which it is embedded.

The concept of "meta" is also used extensively in the context of IS modelling.

- Meta-data: data about data
  e.g. used in data warehousing, CASE tool data dictionaries, database administration and XML specifications. Some examples of meta-data standards are the Dublin Core (web resource discovery), GILS (government information) and FGDC (geographic data sets).
- Meta-modelling: a model of a model, modelling language or modelling process
  e.g. used for explaining or defining modelling concepts, for developing very high-level models, for the design of pure OO tools and languages, etc.
- Meta-CASE: a CASE tool to create CASE tools
  e.g. MetaEdit+ [MART96], Alfabet, Objecteering, Paradigm Plus, sBrowser [BEZI98]

The following explains the connections between a model and its meta-model. A *model* is a collection of artefacts assembled during modelling of a system such as a software system. Typical concepts found in a model are, for example, "Customer", "Order", "StreetAddress", but also "default is 5.7 seconds" and "Z may never happen before X and Y have both happened".

A *meta-model* is an information model for the information that can be expressed during the modelling process or in the model. Typical concepts found in a meta-model are, for example, "Class", "Process", "Constraint" or "Method".

Finally, in order to create a meta-model, one needs a language in which this meta-model can be expressed. The *meta-meta-model* is that language. The reason for its name is that a meta-meta-model relates to a meta-model the same way was a meta-model relates to a model.

In a number of cases, the meta-level and lower level are not clearly separated. This is not only the case in many KR languages but also in a number of "pure" OO languages where a class of classes is treated in a similar way as an instance of that class [GEIS98a].

The distinction between meta-model and meta-meta-model is reflected in the four-layer architecture in Table 2-3, as in e.g. [OMG99; GEIS98; BEZI98a]. This conceptual framework for meta-modelling explains the relationships between meta-meta-model, meta-model, model and (not entirely correctly named) "user data". Together they form four layers on top of each other.

Table 2-3: Four-layer architecture of meta-models.

Level	Type	Example contents
M3	Meta-meta-model	"MetaEntity"; "Package" and "MetaRelationship"
M2	Meta-model	"Process"; "Class"; "Method" and "Attribute"
M1	Model	"Customer"; "Calculate.Pay"; "OrdersFrom"; "Account.Balance" and "Employee.Name"
M0	Data	"Invoice # 92432"; "US$500" and "John Doe"

The meta-meta-model also requires a (modelling) language to express its model. Hence there is a risk of infinite regress. In practice, a meta-meta-model typically uses a very small number of "intuitive" modelling concepts e.g. three [LEME98], four [NISS95] or five [VANH99]. Alternatively a meta-meta-model defines itself in a recursive nature, as has been done in UML and MOF:

> "For a meta-modeling framework to be useful, some way must be found of terminating the meta-hierarchy. [...] This submission borrows a neat trick from CDIF [...] The trick is to make the entities at the top of the meta-hierarchy instances of other entities at the same [meta-meta-]level, possibly even themselves. Although this has the flavor of creating something from thin air (like particles in the 'soup' of quantum mechanics) it does neatly terminate the meta hierarchy." [PELT00]

[VANH99] discusses two criteria for choosing an appropriate meta-modelling language: consonance and no-loss, which basically translate to a balance between richness and simplicity, representative of the tension between semantic relativism and expressiveness of any modelling language. His theoretical criteria appear to be ignored by practitioners who seem to prefer to use the relatively complex and rich UML class diagram / OCL combination for their meta-models (see below).

An interesting observation from meta-model analysis is that the strict distinction between concepts and relationships is actually quite spurious. What is considered a relationship in one context can easily be constructed as an entity in another and vice versa [RUDL96]. This is true on the low level for any relationship that can carry attributes e.g. if a customer orders a product, the order will normally become an entity in its own right. Similarly, an association between two entities can legitimately be viewed as an attribute of any of the two entities or become an entity itself [HAMM90; AVIS95]. As an example, the manager of a business unit has been modelled as a relationship between the entities of managers and business units, as an attribute of the entity business unit, or conversely, the business unit that is being managed could be an attribute of the manager entity. The object-oriented approach has not solved this issue because there are several ways of modelling the "role" concept [MARS00]. At the model level it is clear that what is a flow (relationship) in a DFD often becomes an entity in an ERD and the process (node) in the DFD is often reflected via a relationship in an ERD. Similarly, a message (relationship) between classes in a UML sequence diagram would have to become an informational entity when modelling the physical network, whereas the nodes on each entity life line are in fact processes which could be modelled as relationships in that entity's UML statechart-diagram.

Since meta-models are models, generic modelling tools can be used to develop and document them. This will be the case if an existing modelling language such as the UML is used as the meta-meta-language. Specific meta-modelling tools such as Meta-Edit, ConceptBase, MethoModeller or ToolBuilder (see above) and meta-modelling methodologies such as MEMO [FRAN99a] or GOPPR [VANH99] have also been developed.

2.11.2 Why Use Meta-Models

There are a number of situations for which meta-models are highly useful [VANH99; FRAN98a]:

- Meta-models can serve as conceptual schemata for repositories that hold software engineering and related data as well as to develop flexible modelling software such as CASE tools.
- Meta-models have been used to define certain modelling languages e.g. IDEF and UML. This is also useful in defining the extensibility of a language.

- Meta-models are an essential part of technologies (together with a transfer mechanism such as a file format) that allow interoperability of modelling tools. This is illustrated by their use in the model interchange standards CDIF and XMI.
- Meta-models can be a tool to help to understand the relationships between concepts in different modelling languages, since a meta-model identifies a usually small number of core concepts.
- Meta-modelling can be used as a general technique for integrating and comparing models from different domains, but also to identify overlaps between models from the same domain [GEIS98a]. This research will examine the meta-models of the different enterprise models as part of its model evaluation framework.

As an example of practical meta-model use, imagine a systems integrator faced with the challenge to integrate multiple modelling and other tools in such a way that they more or less appear like one tool to the user. In order to do this, she needs to know what tool supports which concepts, and how each concept of each tool relates to the concepts of another tool. A meta-model analysis provides a list of concepts and their relationships. The resultant meta-model can be used to decide which data is to be kept where, and how tools are supposed to talk to each other.

2.11.3 Types of Meta-Models

There are a number of different types of meta-models:

- **Process**. Process meta-models describe the modelling process or activity e.g. how to go about analysing and modelling an enterprise. These are used e.g. in methodology engineering and evaluation see, for example, [VANH99]
- **Model.** By far the most prevalent meta-model, however, is concerned with meta-modelling the *product* of a modelling activity: it is a model of a model. This category also includes most meta-meta-models. This category can be subdivided into three types of meta-models: those that describe the modelling notations or symbols ("rectangle with rounded corners, arrow with a diamond head"), those that define the modelling concepts that are used ("class", "relationship") and those that describe both.
- **Domain.** Some meta-models describe the modelling domain using high-level, abstract concepts e.g. an enterprise meta-model could introduce the concepts of resource, agent, goal, plan etc. Many authors consider this as a high-level model rather than a meta-model, hence my earlier suggestion of the name "mesa-model" for this type of model. The distinguishing attribute of a domain meta-model is that all the entities in the "mesa-model" are *abstract* classes.

Most meta-models are based on the semantics of the underlying model elements. However, it is also possible to take a purely syntactic stance and derive a meta-model purely based on the generalization of syntactic similarities e.g. according to attribute domain rules, allowable values or data types. This applies especially to the "domain meta-models", where it increases the flexibility and usability of the derived models. This is analogous to the practice in domain modelling of generating super-classes based on syntactic rather than semantic considerations:

> "Although Santa Claus is not likely to be considered in the same semantic category as the Eiffel Tower, theoretically, we could build a data model where they are both valid examples of the same entity." [HO99]

2.11.4 Some Examples of Meta-models and Meta-Meta-models

The following are some representative examples of publicly available meta-models for modelling notations:

- **DFD:** an example of a meta-model of the DFD using NIAM as the meta-modelling language is given by [TERH96].
- **UML:** OMG has relied extensively on meta-models to specify each of the UML 1.3 modelling constructs [OMG99]. As an example, the UML meta-model for the class-diagram model element of "relationships" is given in Figure 2-3. Note that the UML meta-models are also drawn using the UML itself: they use UML Class diagrams to define the meta-entities and OCL to describe their well-formedness constraints [GEIS98b].
- **CDIF:** CDIF has been defined by the CDIF Division of the EIA, an industry standards committee, by means of meta-models. CDIF is also being standardized at an international level through ISO/IEC JTC1/SC7/WG11. See http://www.cdif.org.
- **PROMPT:** This frame-based KR model has a meta-model consisting of classes, slots (attributes), facets (named ternary relations) and instances (members of classes) [NOY00].

Note that meta-models are not necessarily limited to describing other modelling techniques: the HL7 v.3.0 (Health Level Seven) clinical data interchange format prepared by the JWG-CDM [http://www.mcis.duke.edu/standards/HL7/]; a proposed Data Element Registry model [GILL97a]; and the LTM - Legacy Transition Meta-model for the repository of current and target IS [http://www.systemtransformation.com] have all been defined using meta-models.

A large number of meta (or *mesa*)-models for enterprise *domain* modelling can be found in the literature. Here are some examples:

- Marshall's meta-model [MARS00] was already given in Figure 2-2. Note that Marshall extends UML with the following four enterprise stereotypes to facilitate enterprise modelling: <<purpose>>, <<process>>, <<entity>> and <<organization>>. He uses these as high-level (abstract) business objects in his BOMA model.
- The basic meta-model of business modelling concepts according to [ERIK00] consists of the following entities (the relationships between them have been omitted): problem, goal, process, event, state change, interface, rule (constraint, derivation or existence) and resource (information or thing, with thing being physical (e.g. people) or abstract).
- The meta-model for the ODP enterprise viewpoint, as given by [STEE00], uses the concepts of behaviour, action, role, objective, community, contract, actor, artefact, principal, policy, constraint and object. His meta-model is also in UML.
- [LOCH98] employs the following four meta-classes: LEGAL ENTITY (players, actors); COMMODITY (products, services, resources); EVENT (process steps, acts) and INSTRUMENT (objects, tools). Note the limited correspondence to Marshall's meta-model. However, there is a much closer correspondence with the four primitive classes of PSL: Object, Activity, Activity_Occurence and Timepoint.

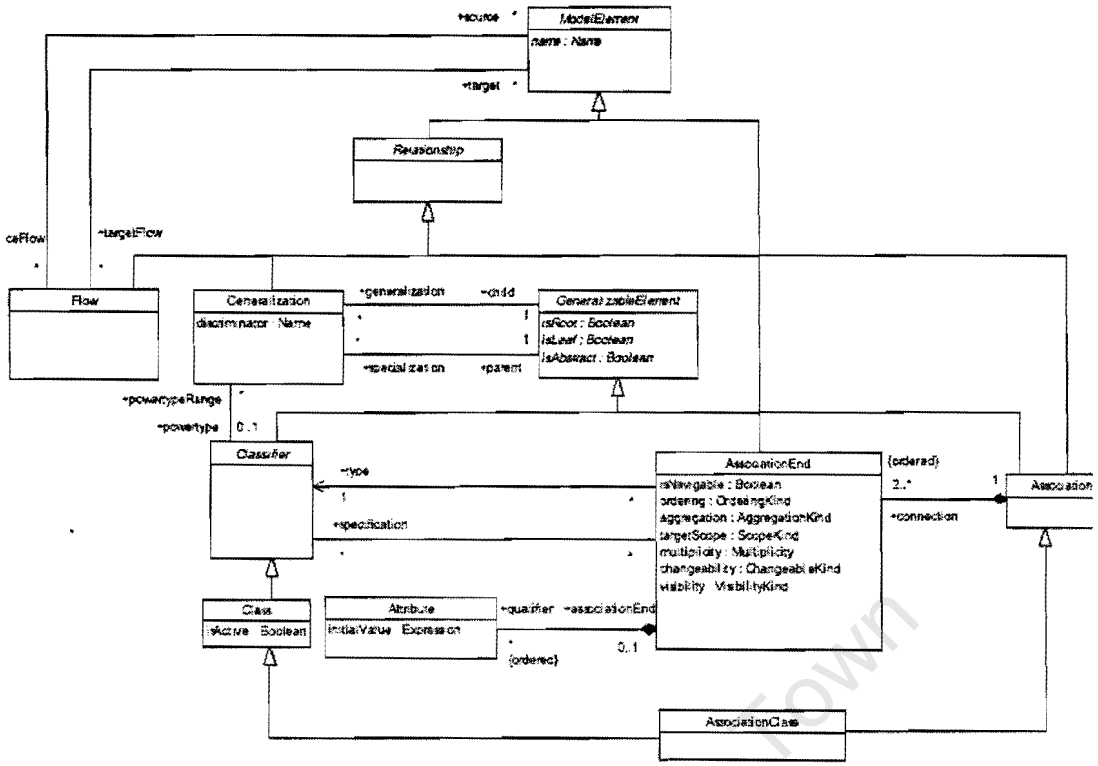


Figure 2-3: The UML meta-model for "relationship" [OMG99].

Finally, there are a number of meta-meta-models. It is interesting to note the commonality between the meta-meta-constructs:

- [MANS96] describes the meta-meta-model for a CASE repository which consists of nodes, links, groupers, diagrams and properties.
- [GERB96] describes a very generic meta-meta-model for any knowledge representation system based on conceptual graphs theory. The following constructs are used: concept (specialized into individual, generic and universal concepts), conceptual relationship, conceptual graph and the definition of concept type, relation type and cardinality.
- [NISS95] uses the following four constructs suffice: nodes, links, assertions and classification.
- [LEME98] uses a meta-meta-model with only the three concepts of "concept", "relationship" and "modularity" to compare the MOF, CDIF and sNets formalisms.
- [FRAN98a] similarly compares the three meta-meta-models of MEMO, CDIF and UML but uses a total of 27 different meta-meta-model concepts in his comparison. Many of these concepts, however, appear only in the UML meta-meta-model and Frank, quite correctly, expresses some reservations about their appropriateness at the meta-meta-level.
- A very detailed meta-meta-model (several hundred pages long) can be found in *Platinum Technology*'s submission for an Object Analysis and Design Facility to the OMG/OA7D RFP (see http://www.planinum.com/clrlake).
- The Meta Object Facility (MOF) as described by OMG is also in the form of a fairly detailed meta-meta-model, described by a combination of UML notations, CORBA interfaces and OCL [GEIS98b; CRAW97]

An obvious question is whether any meta-meta-meta-models and even higher-order models exist. Generally, infinite regress is avoided by defining meta-meta-model concepts in their own language. Although this introduces the problem of tautological or reflexive definitions, in practical terms the meta-meta level concepts are seen as intuitive and the benefit of closure is seen to outweigh any philosophical problems.

This first section of the literature survey focused on the nature of modelling. The application of modelling theory to organizations has been pursued in a number of different sub-disciplines of IS. These are investigated in the next chapter.

Chapter 3: Reference Disciplines for Enterprise Modelling

Having reviewed the nature of modelling, this chapter will deal with the current state of enterprise modelling across different reference disciplines. Enterprise modelling is an interdisciplinary field of research. This section aims at identifying the more important reference disciplines and the contribution they make to enterprise modelling. In addition, a large number of more specialised research areas in IS are concerned with the modelling of the enterprise. These fields, which will also be described briefly, are practitioner-driven. The disciplines are listed in an order reflecting a roughly decreasing theoretical focus i.e. from the philosophical and hard sciences to more applied research.

Due to the enormous scope of this section, and since the reader is assumed to be more familiar with the IS-related disciplines than the others, relatively more emphasis will be placed on the fields further removed from IS and IT. For example, ontologies and systems theory are discussed in more detail than systems engineering. Readers familiar with the various reference disciplines will be able to skim the text lightly or skip entire sections altogether.

3.1 Systems Theory

Systems theory is hard to define succinctly. Authors spend many pages describing what systems are before discussing the various general approaches in systems theory [CHUR68; SKYT96]. Others avoid the issue altogether [CHEC92]. In a sense, this reflects the nature of systems theory: it is a methodological approach to study phenomena rather than one single consolidated and coherent body of knowledge. Systems theory applies to a large and heterogeneous set of theories, whose only common denominator often is the use of certain basic terminology and the general methodological approach [VONB81; CHEC81].

[HEYL92] ventures the following succinct definition of systems theory:

> "the transdisciplinary study of the abstract organization of phenomena, independent of their substance, type, or spatial or temporal scale of existence. It investigates both the principles common to all complex entities, and the (usually mathematical) models which can be used to describe them."

The literature appears to make no distinction in practice between systems theory and *general systems theory (GST)*.

Since systems are everywhere an observer or researcher wishes to find them (refer to Boulding's aphorism "a system is anything that is not chaos" or Weiss' "a system is anything unitary enough to deserve a name" [SKYT96]), the discipline is as diverse as can be expected. It is meant to be a "science of everything". It comes as only a small surprise that, since it is classified as "000" under the Dewey Decimal System, it shares the same library shelves as the philosophy of science and theory of knowledge. Unfortunately, the fact that the same shelves also house witchcraft, ufology and ESP, seems to trouble some scientists.

The contributions of systems theory towards enterprise modelling are quite substantial.

3.1.1 Holism, Complexity and Other Systems Concepts

A substantial number of key concepts of systems theory have infiltrated the standard *vocabulary* of the enterprise modeller: multilevel systems hierarchy, open and closed systems, system environment, feedback and feed-forward loops, negative and positive feedback, control mechanisms, reflective

goal-seeking systems, black-box versus white-box, emergent properties etc. as explained in [VANB99a] or in any standard text on systems theory.

Apart from a whole vocabulary of systems concepts, the study of *complexity* is a sub-discipline of systems theory. Complexity theory is concerned with defining what complexity is, how one can measure it and what modelling challenges it poses. Some of the implications of complexity on modelling can be found in [STER00; AGAR99; FUNE01]. Unfortunately, little of the more advanced theoretical contributions has found its way into practical applications such as e.g. complexity metrics [EDMO99].

A related important conceptual contribution is the necessity of a *holistic* approach towards modelling the enterprise [SKYT96]. The enterprise is a complex system and, unlike many artefacts in the engineering disciplines, its behaviour is unpredictable due to the presence of independent and conscious actors (humans and social groups), interrelationships with its dynamic environments, the presence of many non-linear relations and the many internal and external feedback loops present in the system. In fact, organizations are often more usefully viewed as a web of relationships between various stakeholders - a "nexus of contracts" - or as devices for meeting individual and group needs [NICK00]. The inherent and often unresolved sociological complexity (the "messiness" present in many social systems) was (and sometime still is) often not acknowledged by the "hard system" way of thinking of computer scientist and software engineers who fail to understand that the domain of IS modelling is a fundamentally different qualitative nature [CHEC98]. The more recent work in complexity theory and chaos theory merely emphasizes the point.

3.1.2 Soft Systems Methodology and System Models

The best known "mainstream" contribution of systems theory is the SSM (Soft Systems Methodology). This approach has been advocated strongly by, amongst others, Checkland [CHEC81; CHEC98] and is pursued vigorously at a number of British research institutions. The main tenet of SSM is that IS is concerned with building models from human/social artefacts or organizations. These *soft systems*, unlike the physical world of the engineers and natural scientists, have no hard laws and therefore a human-centred methodology needs to be adopted. The methodology, developed from action research, pays particular attention to how subjective observers ascribe meaning to data. Rather than laying down a strict and rigorous set of rules and procedures, the SSM is often illustrated by means of case studies [HIRS97].

Systems theory has also contributed a number of modelling notations for use in enterprise modelling. Three well-known techniques are the stocks and flows model (dynamic modelling), the human activity model (SSM) and Beer's viable system model.

A widely used technique for modelling dynamic systems is the stock and flows model. This technique gained immense popularity due to Forrester's famous report [FORR71] and an excellent exposition on how to apply the technique to dynamic enterprise modelling can be found in [STER00]. In standard graphical systems modelling notation, rectangles represent stocks, pipes and valves represent flows between stocks; arrows indicate causal influences between system indicators or variables (can be positive or negative); and special symbols are used for time lags, reinforcing and balancing feedbacks, in/output from/to the environment etc. An advantage of system dynamics models is that they can include "soft variables" that are essential to the explanation of a system but can not be measured by means of hard data. Other strengths are the almost natural model validation by means of the implied simulation capability, the intuitive specification of system structure, the distinction between

correlation and causality, and the easy incorporation of non-linear relationships [ALFE94]. Some of the technique's principles have also been incorporated into the IDEF modelling suite.

The SSM style *human activity model* is a simple, very intuitive diagram with the following main components: a set of activities, linked by means of dependency arrows, within a boundary forming an operational sub-system. Moderating this subsystem is a separate monitoring and control subsystem [CHEC98].

Also used extensively in SSM is the *rich picture diagram*, a cartoon-like summary of a complex situation akin to a pictorial brainstorm. It is *not* intended to be a formal system description but rather a summary of all known information, including soft facts such as opinions, hunches, synergies, and interpersonal relations. Its three major components are: elements of structure (all relatively stable components); process elements (changing elements or transformations) and relationships between any of the above [DAEL94].

Beer developed a theory and accompanying set of laws which govern any viable system, and applied them to a great number of organizations, ranging from small not-for-profit organizations to major countries. His approach could be classified as organizational cybernetics: it sees the organization as a system striving to maintain an active and dynamic balance (homeostatis) within its environment. To aid his analysis technique, Beer developed the *viable system* modelling (VSM) technique, a "big brother" to SSM's human activity model, and has made numerous presentations on its application to organizational modelling, see [BEER85]. Some characteristic features of the VSM are its many control loops (e.g. local regulatory control, sporadic audit, policy monitoring, autonomic regulation), self-reference through lower-level recursion, self-repair and self-awareness. Some special model elements are information amplifiers (e.g. turning a manufacturing decision into an operational schedule), attenuators (e.g. condensing sales figures into a summary report), and transducers (en/decode messages when they cross boundaries). Other essential parts of the VSM are the principle of homeostasis (systems operating within a basin of relative stability, maintaining a steady but dynamic state of operation), the concept of requisite variety (to control a complex system, you need a control structure which can handle at least that level of complexity - hence the use of amplifiers and attenuators) and various principles relating to the amount of information that a channel can carry and agents can process.

Other systems theoretical methods have made only a minor contribution. An example is the virtual life / genetic algorithm approach which uses generations of huge populations of virtual enterprises, modelled by means of a set of fairly simple organizational characteristics - the enterprise's genes - to determine the optimal organizational structure under various different contexts or environments [WATK98]. Another example is the use of " holons" (any entity which is at the same time a whole onto itself and a part of other wholes) as modelling units e.g. the holonic-based process modelling [PRES97b] where a factory could be specified as five levels of holons: facility, shop, cell, workstation and equipment.

3.1.3 Specific Enterprise Models from Systems Theory

The philosophy underlying SSM precludes its use for generic enterprise models: each organization is unique and the importance of the human element guarantees diversity in models. Hence SSM models are ad-hoc and limited to particular situations or cases:

> "Soft systems thinkers [...], in contrast to hard systems thinkers [...] do not try to design models for use over and over again. This is not seen as productive because of the widely different viewpoints which will be relevant

> in each problem situation. Instead, what is usefully replicated, as Checkland argues, is the methodology employed." [JACK97]

Although Beer's VSM can be seen as a high-level model, no instance of a more detailed instantiated version for "generic" use could be found.

Despite the fact that the dynamic business analysis could easily be the source of enterprise models, none could be found in the literature surveyed. This does not imply that they do not exist: at least the Manufacturing Systems Integration Research Institute of the University of Loughborough has a fairly elaborate though so far unpublished model [as seen on a personal visit, July-1998].

The most seminal research on generic enterprise models is found in Miller's "General Living Systems" [MILL78], which looks at common elements, processes, indicators and laws pertaining to "living systems" at eight different levels, from the living individual cell to supranational systems with the organization falling neatly in the middle. He does a detailed analysis of the nineteen (in 1990 expanded to twenty) subsystems for each of the levels. Each of these subsystems is identified as critical for living and reproduction. His analysis is done mainly by means of examples and detailed investigation of the applicability of the laws, supported by a large number empirical studies. Much of the analysis is related to the flow and use of information although all of the analysis is in informal notation. The strength of Miller's model is that it not only applies to organizations, but it can also model communities, countries, or biological systems. Hence it is not only the most theoretical but also by far the most universal of all enterprise models reviewed here.

Unfortunately, Miller's work never seems to have entered mainstream IS. However, it is interesting to note that [FERR95] "re-discovered" the similarity between biological systems and organizations:

> "[I found a great] similarity between the interactions and control systems operating within biological systems and those inside a business. The parallels are numerous and can provide valuable insights into solutions for some of today's more difficult business-related problems."

The term "living systems model" has also resurfaced, albeit in the slightly different connotation of a continually updated model reflecting the changes of the underlying organization [WHIT97a].

3.2 Formal Ontology Research

Ontologies are commonly associated with philosophy, where ontology studies theories about the nature of existence and the types of things that exist [BERN01]. However, the study of *formal* ontologies is a branch of artificial intelligence, particularly useful for knowledge engineering, natural-language processing and knowledge representation. Ontology researchers use major inputs from philosophy, logics, linguistics and cognitive sciences [GUAR95a]. The field has received additional impetus by web researchers looking for formal ways of defining and interchanging terminology, e.g. in e-commerce and cooperative information systems, as well as by researchers in knowledge management and intelligent information retrieval/integration [FENS01a; FENS01b].

3.2.1 Definition of Formal Ontology

As can be expected in an interdisciplinary field, there are a large variety of definitions as to what a formal ontology is: from taxonomic class hierarchies to formal vocabularies and constraining axioms defined in first-order logic (FOL) [CIOC00]. A good overview of the range of definitions and a detailed analysis of them is given in [GUAR96; GUAR97a]. The most commonly cited definition is Gruber's "an ontology is an explicit specification of a conceptualization" [GRUB93a] with the conceptualization referring to the ontological commitment within the domain, and the specification

referring to the representation of the knowledge [VALE96]. However succinct the definition, it seems to lack precision and clarity (are a written-down travel plan of a proposed trip or a blueprint of a house also ontologies?), and the following appears to be a more detailed definition:

> "An ontology is a formal explicit description of concepts in a domain of discourse (classes, sometimes called concepts), properties of each concept describing various features and attributes of the concept (slots, sometimes called roles of properties), and restrictions on slots (facets, sometimes called role restrictions). An ontology together with a set of individual instances of classes constitutes a knowledge base. In reality, there is a fine line where the ontology ends and the knowledge base begins." [NOY01]

Although this seems far removed from the philosophical concept of ontology, it is in fact a specialization since a formal ontology is a description of the kinds of things (physical or conceptual) that make up a domain [PRES97b]. A very useful set of descriptions of related terms is given by [BEZI98a]:

- **Vocabulary:** a language dependent set of explained words, not claiming any universality or formality.
- **Data dictionary**: a technical form of vocabulary embedded in a computer system.
- **Data base schema**: defines aspects of a database structure including attributes and their domains.
- **Taxonomy**: a hierarchy of concepts, linked by *isA* (not *instanceOf!)* and possibly *part of* relationships.
- **Ontology**: an explicit and precise description of concepts and relations within a particular domain e.g. an organization, study field, application area.
- **Mereology (or mereotopology)**: an account of parts and wholes in a given area.

Despite these distinctions, most researchers agree that a taxonomy forms the main backbone of any ontology, which is then expanded in more detail by adding attributes and relationships [GUAR99a].

3.2.2 Uses of Ontologies

The ontology literature is full of motivations for and uses of ontologies. Basically, ontologies explain the structure of knowledge and facilitate the sharing of knowledge [CHAN98]. The following is a succinct list of ontology uses [NOY01]:

- To share a common understanding of information and its structure, among people or software agents.
- To make domain assumptions explicit.
- To analyse domain knowledge, and separate it from operational knowledge.
- To enable the reuse of domain knowledge.

A more elaborate discussion is found in [USCH96b]:

- Communication between humans from different backgrounds, especially by reducing ambiguity and integrating different user perspectives.
- Inter-operability: an ontology can serve as an *inter-lingua* between different systems using different concepts or vocabulary, replacing the necessity of n x n translation pairs to n x 1 (to the ontology) necessary translations.

3.2.4 Ontology Languages

Because ontologies require formal notations, a number of ontology-specific languages have been designed. Typical logic-inspired languages are FOL, Ontolingua, LOOM, Frame-Logic, and description logics [FENS01b]. However, the popularity of web-based cooperative ontology editing tools has prompted a new wave of XML-based languages: SHOE, XOL (Ontology Exchange Language), Ontology Markup Language (OML and CKML), and OIL (Ontology Interchange Language) [BERN01]. These newer ontology languages are not just of sole interest to ontology researchers any more, since they have become a practical approach to the problem of knowledge representation on the web [VANH99].

The formality of the languages tends to frighten away enterprise modellers, although modern modelling tools allow the export and import of models into a number of formal languages e.g. CDIF or UML/OCL: "The modeller will soon discover that, when dealing with 'subject areas', she/he is working with ontologies in everyday life, a surprise much as Molière's Monsieur Jourdain has been 'astounded to find that what daily issued from his mouth was such an important thing as prose' " [BEZI98a].

3.2.5 Enterprise Modelling and Ontologies

A large number of ontologies have been developed, including the mammoth CYC project [LENA90], upper ontologies such as the Penman Upper Model [BATE90] and its derivatives Pangloss and the Revised Upper Model, the MicroKosmos ontology [GUAR99a] and the linguistic ontologies WordNet, Kactus and Plinius [USCH96b]. But a substantial amount of ontology research is specifically concerned with the enterprise. [FOX98; WHIT97a] give the following progress report on efforts underway or completed at that time:

- **ICAM GEM:** An effort from the US Air Force to create a standard model and semantics for the aerospace industry
- **TOVE:** The Toronto Virtual Enterprise project from EIL, driven mainly by Fox & Gruninger, includes detailed formal ontologies for enterprise resources, costs, quality, organization structure, product and agility [FOX93b; GRUN96b; GRUN96c].
- **Enterprise Ontology:** The AIAI Enterprise Project completed its ontology in 1997 with a conversion of its semi-formal natural language ontology into Ontolingua. Apart from generic activities and processes, its focus areas are organization (structure), strategy and marketing [USCH96a; USCH98].
- **IDEF ontologies:** provide a rigorous foundation of the IDEF0 and IDEF1x as tools for modelling enterprise [WHIT97b].
- **PIF & PSL.** The Process Interchange Format has been defined as a formal ontology to provide a common format for the exchange of process information between various tools (workflow, simulation, BPR ...). The Process Specification Language (by NIST) has a purpose similar to the PIF but goes further to enable the exchange between manufacturing applications, including real-time product realization and project management [KNUT98; SCHL99; SCHL00].
- **CIMOSA, PERA and GERAM:** These are three frameworks derived from the field of enterprise engineering and enterprise integration (see below) whose descriptions and formalizations are ontology-based [BERN94; BERN96a; BERN96b].

Only the first three can be considered to be comprehensive and fully populated enterprise models; the others are more confined to a specific sub-area of the enterprise, or focus mainly on modelling notation. An important omission from the above list is the ontological work around the STEP (ISO 13030) product specification standard [GUAR97b; METZ96]. The Workflow Management Coalition is also working on an ontology-based reference model [WHIT97a].

Presley has made a valiant attempt to integrate all of the above ontologies in his doctoral research [PRES97a; PRES97b] with the specific aim of addressing the following shortcomings of current enterprise modelling efforts:

- The inability of current modelling methods to consider multiple enterprise views at the same time.
- The poor explicit formal support for selection of resources during enterprise design, due to the lack of integration of process and resource views.
- The problem of model obsolescence and lack of model reuse, due to the poor support of incremental model change.

Both TOVE and the Enterprise Ontology will be used as representative enterprise models for this research. The methodologies that were followed to develop TOVE and AIAI Enterprise Ontology are discussed and evaluated against/mapped to the IEEE Standard 1074-1995 (for software development) by [FERN99] and summarized in [GOME99]. GOME99 also confirms that TOVE and AIAI are the only two enterprise domain ontologies.

In addition to the "academic" enterprise ontologies, a number of "commercial" ontologies are currently being developed to address the need to interchange commercial data between heterogeneous systems on the Internet: UNSPC (www.unspsc.org), RosettaNet (www.rosettanet.org) and DMOZ (www.dmoz.org).

Apart from the design of enterprise-specific ontologies, ontology engineering has contributed the following to the discipline of modelling:

- It has generated a large volume of fundamental research on *knowledge representation* and *knowledge engineering*.
- Ontology researchers have designed a number of *frameworks* for the evaluation, comparison and integration of ontologies. Many elements from these frameworks are also very useful when comparing and evaluating enterprise models.
- A lot of ontology research focuses on *upper-level* concepts i.e. the equivalent of the *meta-level* in modelling.
- Ontology researchers have developed a number of *tools* for the development of ontologies. Although the earlier tools required a solid understanding of FOL or an equivalent logic language, more recent versions are much more graphically oriented. Ontology editors are possible candidates for model management and meta-modelling.

3.3 Enterprise Integration and Enterprise Engineering

A "hard science" approach to enterprise modelling originating from the engineering sciences is the discipline of enterprise engineering and the associated goal of enterprise integration. This approach originally focused on manufacturing enterprises. Of lately, the focus has shifted to the *virtual enterprise* which, in this context, is defined as an integrated but flexible set of processes that can be reconfigured dynamically to ensure optimal operation by means of control engineering principles and analysis techniques.

[JEUS01]. Note that EI models have the additional requirements of full model semantics i.e. formal model representation and model constructs that enable model execution (or model *enactment*). In model-driven enterprise integration, process models are "live": the model and execution are tightly connected and a modification of the model can immediately and automatically change the way the (usually manufacturing) process is executed [BRUN97]. [WHIT97a] makes a strong case for living models and defines a living enterprise model as "a model which drives, and is driven by, the daily operations of the enterprise".

The need for executable enterprise models has prompted the development of formal specification techniques, such as the Process Specification Language (PSL), as well as enterprise ontologies (refer to the section on ontologies). The focus on the semantics of enterprise models has prompted a number of fundamental papers on the nature of enterprise models [e.g. BERN96a; NELL96; GRUN96b]. In addition, a number of standard enterprise reference models have been developed [WILL98]. Some of these reference models will be part of the data sample for this research: the TOVE ontology by the Enterprise Integration Laboratory (University of Toronto) [FOX93a; FOX93b; FOX98] and the Purdue Reference Model for Computer Integrated Manufacturing [WILL91].

In addition, substantial effort has been invested in the evaluation and development of the appropriate methodologies for EE. [BERN96b], as mandated by the IFAC/IFIP Task Force on Architectures for Integrating Manufacturing Activities and Enterprises, reviews a large number of methods but with a particularly detailed analysis of GRAI-GIM, CIMOSA and PERA. An important criterion of the authors is that of full enterprise life-cycle support i.e. the methodology must support the design, construction, development, operation and shutting down of the *enterprise*.

3.4 IS Methodologies and Method Engineering

3.4.1 IS Methodology: Definition and Motivation

"Over the past twenty years or so there has been a considerable growth of interest in IS development methodologies" [PROB01]. IS researchers have to live with the unfortunate coincidence that the scientific research term of "methodology" (defined in the Oxford Dictionary as "a systematic method of scientific research") is simultaneously used in a completely different sense in the IS field, namely referring to a structured method in which system analysis and design (including to a large extent modelling) is conducted.

A full discussion of this tension as well as a detailed analysis of the different meanings and definitions of IS methodologies is given in [JAYA94]. In the end, Avison & Fitzgerald's definition of an information systems development methodology appears to be the clearest:

> "A collection of procedures, techniques, tools, all documentation aids which will help the systems developers in their efforts to implement a new information system. A methodology will consist of phases, themselves consisting of sub-phases, which will guide the systems developers in their choice of the techniques that might be appropriate at each stage of the project and also help them plan, manage, control and evaluate information systems projects." [AVIS95]

The distinction between a methodology and a modelling technique seems clear-cut: the technique is a way of representing a model (in terms of the meaning triangle of fig 2.1: mainly concerned with V3) as opposed to a methodology which is concerned with the entire process of arriving at a model (using the meaning triangle: how to get from V1 to V3) as well as the subsequent implementation in an operational system. Despite this, even practitioners often use the terms method, methodology and

model technique almost interchangeably, not a trend applauded by the academic researchers [JAYA94]. It is suggested that "*method*" should be used as the most generic term and the use of term *methodology* should be clearly separated from the concept of modelling technique.

The motivations for the development of methodologies are largely similar to the reasons mentioned for modelling: standardization, quality assurance, productivity increase, maintenance issues, metrics, professionalism, tool support etc. [MCLE92].

Some of the *objectives* of a methodology are the following [AVIS95]:

- Specify the system requirements accurately.
- Enforce systematic system development thereby enabling project monitoring and control.
- Provide a system at acceptable cost and within a reasonable time frame.
- Minimize the impact of requirements changes during the development process.
- Document the system and allow for future maintenance.
- Provide a system that ultimately adds value to the various stakeholders.

There are many methodologies: in 1994 [JAYA94] estimated (perhaps rather generously) that there were over 1000 brand-named methodologies in use all over the world, a huge increase from only about 300 in 1988. There is every reason to suspect that this number has since increased even further [PROB01]. The "methodology jungle" [AVIS95] is partly reflecting the wide variety of modelling notations (most notations have at least one method supporting it) but mainly due to the plurality of the IS field as reflected in the many academic approaches as well as variability of domain contexts. The following may serve simultaneously as an example of the variety in methods as well as the explosion in methods. In 1992-93, the OMG commissioned a survey of OO-methods [HUTT94]. Half of the methods, namely Class-Centered Modelling (CCM), Fresco, Marketing to Design (MTD), Object Behavior Analysis (OBA), OGROUP, OSMOSYS, SE/OT, Responsibility Driven Design and Z++, did not make it 7 years later to the CETUS-links for OO-methods, which lists the following mix of both well-known and obscure OO "methods" as of 1st November 2001 [http://www.cetus-links.org/-oo_ooa_ood_methods.html]: BON, *Booch*, BOOM, Catalysis, CBD/e, *Coad/Yourdon*, COMMA, CRC, Convergent Engineering, *Demeter*, DOORS, DOOS, EPA, EROOS, *Fusion*, Goofee, HOOD, IDEA, ION, KISS, MERODE, MOSES, MWOOD, Object COMX, Objecteering, *Objectory*, OEP, Octopus, *OMT*, *OOAD/OOIE*, OOA/RD, OOBE, OOCL, OOHDM, *OORAM*, OOSC, OOSD, OOSE, OOSP, Open, OSA, PAUD, ROAD, ROPES, RUP, Scrum, Skill-Driven Design, SDL, *Shlaer & Mellor*, Softstar, *SOMA*, SOMT, Syntropy, XP. The methods in *italics* are the ones which can also be found in the Hutt's OMG survey. Of the remainder, a number are really only notations (typically 3-letter acronyms e.g. BON, CRC, EPA, ION, SDL) and another few are specifically geared towards engineering applications (e.g. DOORS, Goofee, HOOD).

Many differences between the methods may be superficial, though: "the methods available today display a rich but superficial diversity restricted to the technical aspects of systems but a poverty of other ideas" [STAM95].

3.4.2 Method Engineering

There are a number of issues with the adoption of methodologies. One issue is the "rule prescriptiveness of the method" i.e. the degree to which a practitioner needs to follow the methodology: from a strict cookbook approach to a more flexible toolkit approach. Another is the relevance of any particular methodology to the situation at hand and the way in which the most

appropriate methodology is selected. This second issue relates to several problems: methods are evolving, the learning curve of a new methodology may be steep, tool support may be immature and technological change is fast [MCLE92].

These issues are addressed by the field of *method engineering*, also called *situational method engineering* [ROLL96; BRIN96], which concerns itself with the composing of purpose-built methodologies to suit specific development environments [SAEK95]. Method engineers employ a framework to assess the requirements of the method or task environment and, typically by means of meta-modelling techniques, build methodologies by drawing on existing methods and/or method elements from a method database or method repository, possibly through the use of metrics [MCLE98].

Special tools, such as the sBrowser [BEZI98B], or the meta-modelling tools mentioned earlier (MetaEdit+, ToolBuilder etc.), are used to support the method generation process. These tools are referred to as CAME (Computer Aided Method Engineering) tools [MART96].

3.4.3 Contributions to Enterprise Modelling

The contributions of methodology engineering are in the form of generic high-level data models, the emphasis being on meta-modelling and modelling evaluation criteria.

As intimated above, IT solution providers and consultants employ (often proprietary) *methodologies*. Many times, these methodologies are supported by tools that contain a rich encyclopaedia/dictionary of **high-level data structures** of almost universal applicability (Bytheway 1995). These are then "specialized" or customized to fit the needs of individual clients or customers. The methodologies are typically supported by a specific structural view of the organization and IT, as embodied in an underlying framework.

There is a very close relationship between **meta-modelling** and method engineering: in order to model a method and its deliverables, a meta-model analysis is needed, especially if method tool support is required [GERB96; PAIG97]. All CAME tools rely heavily on meta-modelling techniques and most meta-modelling tools have been developed specifically to aid in method engineering [KELL98; BEZI98].

A final contribution consists of the **frameworks for method evaluation**. These offer a great many valuable criteria (often including various meta-modelling measures) that are applicable to the evaluation of enterprise models, since part of method engineering is the evaluation of the methodology products.

3.5 Software Engineering

The discipline of software engineering has undoubtedly contributed more to enterprise modelling than any other field. Its contributions include the multitude of modelling techniques and methods, the plethora of modelling and CASE tools, the concept of model-driven system design, the paradigm shift from relational to object-oriented modelling etc.

To prevent duplication with previous sections and to provide focus to the discussion, only two specific subtopics in systems engineering will be discussed below: software metrics and analysis patterns.

3.5.1 Software Quality and Metrics

Software quality, "the degree to which the attributes of the software enable it to perform its intended end use" [GILL97a, p.6] or "conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software" [PRES97c, p.185], is an important area of concern in software engineering. In fact, the entire purpose of software engineering is to produce high-quality software [PRES97c]. [GILL97a] indicates some of the differences between software quality and other types of "product" quality: software has no physical existence (and hence its qualities as an intellectual entity are harder to characterize), the user needs are often inadequately known in advance and often change over time, and the underlying technologies change rapidly.

Quality is a composite or multi-dimensional measure, with the constituent elements being assigned different weightings according to the stakeholders and the context. What constitutes the quality of an aeroplane or rocket guidance system will be rated differently from the quality of a computer game. There are many models of quality and most of these will be mentioned in the chapter on the evaluation framework. A software engineering methodology usually has an explicit process for monitoring quality: the quality control (QC) and quality assurance (QA) processes. The ultimate goal of these processes is to ensure not only once-off quality software products, but also to guarantee consistent and repeatable quality software production. Two well-known international efforts at ensuring this are the ISO9000-3 certification and the SEI's Process Capability Maturity Model (CMM) as proposed by Humphrey for US Government software [see e.g. KAN95; FENT96]. Because of the importance of quality software, there are numerous related international standards: [KIT95] lists no fewer than twenty IEEE/ANSI standards governing software quality over a 10-year period!

A critical element in quality control is the collection of software development metrics: efforts at measuring critical characteristics of the software product or the development process [FENT96]. Most literature distinguishes between the concept of software measurement and software metrics (the actual numbers, scales of measurements or identifiable attributes used to measure software). Unfortunately there is no consensus on which characteristics to measure and how to measure them, so there has been an explosion in the types of metrics - at one stage it was joked that there were more metrics than computer scientists [MELT96].

It is important to observe that, where metrics are concerned, the field often suffers from too much rather than too little. One needs to strike a balance between having too many metrics and too few. Too many metrics overwhelm the user and lead to information overload (cannot see the forest for the trees) whereas too few metrics are likely not able to capture the full richness required for full insight. [BASI88].

Another important observation is the need to have metrics available as early in the analysis phase as possible [BRIA94]. This emphasizes the relative importance of metrics that can be applied to models (as opposed to design-level metrics). Briand also emphasizes the fact that metrics in IS are not as absolute as those in the natural sciences – they are designed for specific goals and contexts: "the definition of a metric should be driven by both the characteristics of the context or family of contexts in which it is used, and one or more clearly stated goals that it helps reach." [BRIA94, p.3]

[LAKE94] divides all OO metrics into three major groups: system level, tree-related (refer to inheritance) and class-level (refer to encapsulation). Some metrics belong in more than one category e.g. Lines-Of-Code. This agrees with Kolewe who suggested three system level metrics: number of class hierarchies, number of class clusters (interconnections between classes in a system) and association complexity [KOLE93].

Although there are many typologies for metrics, the distinction made by the famed [LORE94] is most appropriate for this research. Lorentz and Kidd distinguish between project-related metrics (productivity, resource usage, defect density, etc.) and design-related metrics (complexity, size, etc.). The former look at the project management and the development process, whereas the latter are concerned with the quality of the actual system or software product being designed. It is one of the tenets of this research that a good enterprise modelling methodology does not necessarily guarantee a high-quality enterprise model or vice versa. The high correlation between the process and product are acknowledged but, since the development of an enterprise model is often a once-off activity the process of which is not publicly documented, it is difficult to measure the quality of the development process. Hence the importance of the intrinsic product-related metrics in evaluating model quality.

Details of various "intrinsic" metrics and quality characteristics will be listed later in the context of the development and operationalization of the evaluation framework. Unfortunately, most existing software metrics are related to the software product in the later stages of the development cycle (e.g. software code, bugs, testing, documentation etc.) and only relatively few metrics can be applied to conceptual models (e.g. hierarchical depth, complexity, size). Although a number of metric tool suites have been developed specifically for the automatic collection of metrics in software code [e.g. FENT96], most metrics are generated by a special module of the CASE or methodology tools.

Interestingly, [BRIT94] provides a number of different taxonomies for IS metrics. The three "classical" taxonomies suggested are:

- Product versus process (and hybrid) metrics – in this thesis only the end product of the modelling process will be dealt with, so only product metrics would be applicable.
- Elementary versus composite metrics – in the framework of this thesis the focus is on elementary measures. Composite measures or criteria are e.g. "quality" or "usability".
- Objective versus subjective metrics – this forms the basis for the tentative second dimension which is suggested for the framework.

[BRIT94] also suggests a new framework based on the following two dimensions:

- "management categories": design, complexity, size, reuse, productivity, quality and "overall" metrics. In the framework suggested for this thesis, only complexity, size and reuse are applicable.
- "granularity level": metrics on the method, class or system level. For models, only the latter two levels apply.

3.5.2 Analysis Patterns

3.5.2.1 What are Patterns?

Patterns allow programmers to share knowledge about their design. It is based on the premise that programmers encounter many problems that have occurred, and will occur again. Documenting patterns is one way to reuse and possibly share the information about how best to solve a specific program design problem. Unfortunately, patterns have become a bit of a fad and consequently over-hyped [FLOR97].

Since a pattern is a high-level concept, it is hard to define. The following are some definitions from the literature:

> "Simple and elegant solutions to specific problems in OO software design."
> [GAMM94]

> "A named nugget of insight that conveys the essence of a proven solution to a recurring problem within a certain context amidst competing concerns." [APPL97]
>
> "The essential abstraction of a type of solution which balances competing forces in a given context & provides a guide to a suitable solution." [MCLE01]

Patterns can be traced back [JOHN98a] to the architect Christopher Alexander who, in the late nineteen-seventies wrote two books on the use of patterns for building: "A Pattern Language" and "A Timeless Way of Building". In 1987 patterns surfaced again, this time in the context of software development, at the OOPSLA conference. This triggered many papers and presentations, written by people such as Grady Booch, Richard Helm, Erich Gamma, and Kent Beck. This work culminated in the publication of the standard reference work in the field of patterns by the so-called *Gang of Four*: "*Design Patterns: Elements of Reusable Object-Oriented Software*" [GAMM94]. This was followed by many more articles and books e.g. [BUSC96; RISI98; SCHM00 etc.].

An anti-pattern is a pattern that tells how to go from a problem to a bad solution. A good anti-pattern also tells you why the bad solution looks attractive (e.g. it actually works in some narrow context), why it turns out to be bad, and what patterns should be used instead.

3.5.2.2 Common Elements

Although there are a number of "pattern languages", it is generally agreed that patterns should at least contain the following elements, also described as the "Alexandrian form" [BUDI96; COPL98b]:

- Abstract: a short summary of the pattern.
- Name of the pattern: as short and descriptive as possible.
- Problem statement or description of the problem the pattern is intended to solve.
- Context: situation description, including preconditions.
- Solution; often in the form of an OO diagram but usually supplemented with a textual description.
- Examples of the use of the pattern, perhaps with some programming code.
- Rationale for the use of the pattern, with perhaps a reference to possible anti-patterns (see above).
- Related patterns: a cross-reference to other patterns to which this pattern or the problem situation is related.
- Known uses

3.5.2.3 Types of Patterns

There are a number of different pattern types [BUSC96]:

- Architecture patterns: describe the structure of a solution e.g. Model-View-Controller (MVC) pattern, Pipes and Filters pattern, Blackboard pattern.
- Domain patterns: domain-related solution e.g. double entry bookkeeping. Also referred to as *analysis* patterns.
- Design patterns: describe the structure of a solution in technical terms e.g. façade, master-slave, view handler, proxy, forwarder-receiver and publisher-subscriber patterns.
- Programming patterns or *idioms*: these are finer grained solutions in a specific language, e.g. collection patterns in SmallTalk.

The bulk of the patterns in the literature are design and programming patterns. For the purposes of this research, we are really interested in domain or analysis patterns related to enterprises: business patterns such as those described in [COPL96a].

Shelton divides business patterns into three further categories [SHEL96]:

- **Behavioural patterns** describe a repeating set of entity objects and interactions, usually a generic business process such as "process fulfilment" (can be used for customer and supplier orders) or the "dispatcher" pattern which connects multiple service requests with multiple providers [ALFR96].
- **Structural patterns** are recurring sets of business entity objects in type, role or composite structural relationship e.g. the legal party pattern.
- **Semantic patterns** are repeating sets of entity object types and binary relationships that capture definitions, meaning or constraints e.g. the scheduling pattern that can be applied to airline reservations, venue bookings, car rentals, or hospital bed utilization [KENT97].

Two important integrated contributions to the field of analysis business patterns are:

- David Hay who describes a set of standard data model analysis patterns that can be used for standard business modelling situations. These patterns relate to such things as organizations, people, products, contracts etc. His data model patterns use the entity-relationship notation and are specific and detailed enough to be considered as a generic data model for this research [HAY98a].
- Around the same time, Martin Fowler came out with a similar work containing perhaps slightly more abstract but therefore less specific "reusable object models". The work is similar to Hay's except that it uses the OO paradigm:

  > "You can get a good sense of how software technology affects conceptual modelling by comparing the models in this book with those of David Hay. We are both trying to build conceptual models, yet our results are different because he uses a relational technique and I use an object-oriented one." [FOWL97, p.4]

3.5.2.4 Why use patterns?

There are a number of reasons why we use patterns:

- It helps solve practical problems.
- It is a way by which to capture the knowledge of experts.
- A pattern documents design decisions as well as the reasoning behind them.
- It facilitates the reuse of the wisdom of experienced practitioners.
- It shortens the learning curve for novices.
- It helps form a shared vocabulary for problem-solving discussion.
- It shows more than just the solution: it also shows the context (when and where), the design forces at work (trade-off alternatives, misfits, goals and constraints) and how the solution balances these forces.

A less quantifiable benefit of using patterns is that it promotes software quality and aesthetics: "The 'Quality Without A Name' is the quality that imparts incommunicable beauty and immeasurable value to a structure." [APPL97]. An elegant pattern imparts this quality to the design of a software product.

It is important to realize that patterns are not meant to be straightjackets for software design, untested ideas, solutions that have worked only once, highly abstract principles or heuristics or solutions that are claimed to be universally applicable for all contexts.

3.6 IS Architectures and Frameworks

Unfortunately, the IT community uses the terms "architecture" and "framework" in a variety of contexts and with different meanings. Enterprise architectures address three areas: what activities the enterprise performs, how these activities should be performed and how the enterprise should be structured [LILE96b]. It refers as much to "a style or method" of doing things as to a specific set of instructions [WYNS96].

3.6.1 Definitions and Motivation

The following definition of enterprise architecture has been given by one of the best-known gurus in the field:

> "Architecture is that set of design artefacts, or descriptive representations, that are relevant for describing an object such that it can be produced to requirements (quality) as well as maintained over the period of its useful life (change)." [ZACH97]

For an enterprise (or business) architecture, the design artefacts are various models, and the object that is being described is the enterprise. The concept of information architecture is used differently by different authors e.g. [STUR98; FINN98]. Based on Zachman's definition, the information architecture is a component of the enterprise architecture which looks at the information (needs, flows and assets) of the organization. In most of the literature surveyed by [STEV91], it is used synonymously with information systems architecture (ISA). IT architecture is understood to comprise hardware, operating system, application, network and security architectures. Data architecture is sometimes used as just another component of the IT architecture, or sometimes treated as a synonym of information architecture, in which case both information and IT architecture are constituent components of the ISA.

Why is an IS architecture so important? The case for an ISA is often made by an analogy of constructing a large building, road or railway transport system [INMO97]. For instance, in the analogy of a railway, it is important that all rolling stock can use the same gauge tracks, that the couplings between carriages are standardized, that locomotive engines draw the correct power, signals and fuels are standardized, staff skills match the profile of the rolling stock etc. Hence an architecture embodies a vision of how system components should fit together (physically or logically) while maintaining a maximum degree of flexibility, defines the appropriate standards to support interoperability and may include a concrete plan on how to get there from a current less-than-ideal situation. It is clear that there is no such explicit ISA in many organizations: Zachman gives the following statistics for the IS in the average company [ZACH97]:

- 70% of the lines of code are concerned with moving data from system to system.
- 40% of CPU time is spent moving data.
- The average data fact is stored 10 times redundantly.
- In the case of one bank, customer data was spread over 129 different files.

Specific benefits from a well-implemented ISA are reduced long-term costs and risks, higher efficiencies through improved resource (human, data and equipment) utilization, increased flexibility

(agility), more streamlined operations and better decision support. [STUR98] mentions a number of more specific benefits which are worth repeating since they apply equally well to enterprise *models*:

- Facilitates the assessment of off-the-shelf applications for "fit" within the organization.
- Eases the integration of the data and systems of acquired companies into the parent organization.
- Provides a common enterprise-wide language and understanding for communicating across different business areas.
- Speeds the induction of new employees who can use it to orientate themselves by analysis of the appropriate portions of the enterprise architecture.
- Provides a guide for assessing existing data structures and, possibly, their re-engineering with the aim of full integration.

In addition to some of the above, [WYNS96] mentions design simplicity, higher quality systems, reusability, modularization and traceability between solution independent requirements and final implementation.

In order to build an ISA, it is important to have a good understanding and overview of the various IT technology components and options available, and how they fit together. This is the role of the framework: it provides a high-level skeleton that helps to structure the architect's thoughts and generates the available alternatives and areas which need to be addressed: "an integrating meta-model through which concepts, models and methodologies can be structured and their interconnections or differences displayed" [JAYA96]. Hence the architect uses the framework to generate a specific ISA for an enterprise.

3.6.2 Examples of Frameworks

Not surprisingly, since IS researchers excel at developing new methodologies, there are an almost equally large number of *frameworks*: a quick survey revealed more than thirty in 1992 [PERI93].

Most large IT consultancy companies have their proprietary frameworks which they use to guide their consultants, e.g. ICL's OpenFramework which is three-dimensional (3 systems: business, people and technology; 3 aspects: perspectives, qualities and elements; 4 operations: direction setting, modelling, implementing and operating). Proprietary frameworks will not be considered here since they are normally intrinsically linked to a methodology and not publicly documented.

Because many methodologies are framework driven, the distinction is not always as clear-cut as one would like. For example, the GERAM, PERA and GRAI enterprise integration methodologies also use their own specific frameworks to analyse organizations [BERN96b]. The term "reference architecture" is often used for an architecture underlying a specific methodology.

There are a number of "pure" public frameworks i.e. separate from any particular methodology. For instance, [DOVE96] defines what is called a "reference model" but is really a framework for a BPI-like analysis of the degree of implementation of critical business processes and practices for an agile business. This framework uses 24 critical agility areas (e.g. strategic plan, partner relationships, process innovation, knowledge mobilization etc.) which allow the framework to rate a company on a CMM-like scale from 1 to 5 for each of the areas.

However, the two best-known "public" frameworks that are specifically geared towards designing an ISA are the Zachman Framework [ZACH87 & SOWA92] and Everington's Information Framework or IFW [EVER96]. Both frameworks use a two-dimensional grid format.

The IFW is the most recent public framework. Its first dimension is the type of information according to the view that is adopted: the organization view (with strategy, structure and skills as sub-components), the business view (consisting of data, function and workflow components) or the technical view (interface, network and platform). Each of those can then be modelled at various levels of constraint: the deconstruction level (further divided into a domain concepts and domain classification level), the composition level (generic template and design context) and finally the implementation (or operational binding) level. This gives a total of 9 x 5 = 45 cells that need to be considered.

The better know framework is Zachman Framework which, like the pattern movement, has been inspired by the architecture principles of building and construction engineering. Because of its prominence in the IS architecture research, and because of its potential application to model completeness analysis, a more detailed description is warranted. It must be noted that Zachman (and, later, Sowa) initially developed it as an *information systems* framework, although it applies to the enterprise as well[1]. Table 3-1 and Table 3-2 below elucidate the difference between the two.

The first (horizontal) dimension is concerned with the six aspects or views of the enterprise, aphoristically stated as the what, how, where, who, when and why. The second dimension is the scope (confusingly named "views"): from the high-level "ballpark" view to the owner's, the designer's, the builder's, the detailed representation and finally the functional (or operational) system level. This level is roughly equivalent to the level of detail, although it is important to realise that it also shifts perspective: the first row, the scope, relates to the view of the CEO who is looking at the enterprise as a whole, and moving down one row does not only increase the level of detail but also shifts the viewpoint. The two dimensions with six categories each result in a 6 x 6 grid, consisting of a total of 36 cells.

Table 3-1: The Zachman Framework [ZACH97; SOWA92].

	Data (What)	Function (How)	Network (Where)	People (Who)	Time (When)	Motivation (Why)
Scope View	List of things important to the enterprise	List of processes the enterprise performs	List of locations where the enterprise operates	List of organizational units	List of business events/cycles	List of business objectives
Owner's View	Entity relationship diagram	Business process model (physical data flow diagram)	Logistics network (nodes and links)	Organizational chart, with roles, skill sets, security issues	Business master schedule	Business rules
Designer's View	Data model (converged entities, fully normalized)	Essential data flow diagram, application architecture	Distributed system architecture	Human interface architecture (roles, data, access)	Dependency diagram, entity life history (process structure)	Business rule model
Builder's View	Data architecture (tables & columns), map to legacy data	System design, structure chart, pseudo-code.	System architecture (hardware, software types)	User interface (how the system will behave), security design	"Control flow" diagram (control structure)	Business rule design
Detailed View	Data design (denormalized) physical storage design	Detailed program design	Network architecture	Screens, security architecture (who can see what?)	Timing definitions	Rule specification in program logic
Operational View	Converted data	Executable programs	Communications facilities	Trained people	Business events	Enforced rules

[1] In fact, Zachman himself suggested that the framework could be applied to virtually anything, including life itself! And [DEVI01] uses the framework to analyse itself i.e. in a recursive or meta-analysis manner.

Table 3-1 illustrates the type of information that needs to be considered in each of the cells. It is recognized that not many (if any) methodologies address all of the cells of the Zachman Framework. Table 3-2 gives examples of common types of models found in each of the cells, with blank spaces indicating no popular diagramming method yet. It must be noted that a number of diagrams occur in multiple cells: where the diagram incorporates multiple views (e.g. a UML interaction diagram models both data and time) or can be applied at different levels (e.g. a UML class diagram). In other cases, one single cell can contain more than one model artefact (e.g. the UML collaboration and interaction diagrams could appear in the same cell).

Table 3-2: Diagrams associated with the Zachman Framework [DEVI01].

	Data (What)	Function (How)	Network (Where)	People (Who)	Time (When)	Motivation (Why)
Scope View	Business Entities	Vision Features		Vision Stakeholders	Business workflows	Vision Needs
Owner's View	Business Object Model	Use cases	Network configurations	Actors	Use case flow of events	Business rules
Designer's View	Persistent classes	Use case realizations	Deployment model	Boundary classes	Interaction diagrams	Constraints, multiplicities, workflow activities
Builder's View	Data Model	Components	Process-to-node mapping	End user support material	Process model	
Detailed View	Columns, types, keys, indexes	Design classes		UI design classes	State machines	

The Zachman Framework has been applied in a number of different contexts. The following list gives some sample references:

- As the basis for building enterprise information architectures [COOK96].
- As a guide for the implementation of data warehouses [INMO97].
- To inform enterprise modelling methodologies [DEVI01].

The use of IS frameworks is not restricted to high-level architectural designs such as developing an ISA. There are numerous frameworks for developing lower-level system solutions. This type of framework generally consists of class or component libraries which include a comprehensive set of classes that the developer can use to build a solution within a certain business domain or technical context [MCLE01].

Lower-level frameworks typically rely heavily on patterns [JOHN92], although [HAY98b] demonstrates that high-level patterns can be used in the context of Zachman's framework as well. "A software framework is a reusable mini-architecture that provides the generic structure and behaviour for a family of software abstractions, along with a context of memes and metaphors which specifies their collaboration and use within a given domain" [APPL97]. It lacks application-specific functionality, which has to be coded through the use of plug-points. An example of such a framework is IBM's San Francisco framework (and its follow-up WebSphere) which helps JAVA developers with the fast design and implementation of a business system [BENN00]. Another example is SESH's BOMA [MARS00]. Both will be discussed in full in the model section, since they are built around an enterprise model.

3.7 Other Related IS and IT Subject Fields

In addition to the academic fields of research mentioned above, there are other IS-related subject areas that have made a contribution to enterprise modelling. These tend to be more oriented to IT practitioners than to academics. Since the reader is assumed to be familiar with the fields, the discussion will be accordingly brief and focus on their contribution to enterprise modelling.

3.7.1 Data Warehousing

The existence of many separate legacy systems prompted organizations to look at ways of integrating the information. The requirement to have an integrated data view across the organization to facilitate management decision making and enable effective management of an organization's data resource resulted in the need for a data warehousing approach. A typical data warehouse copies, integrates and consolidates historical operational information from various heterogeneous application-specific databases and stores it in a single multi-dimensional database for online access by functional and executive managers [VANB99b]. Hence [PERK96] suggests that perhaps a data warehouse should therefore more appropriately be called an *information* warehouse. Smaller versions, often focused on a particular area, are called data marts.

Related technologies are OLAP (online analytical processing) and data mining. Dr. Codd distinguishes the following four enterprise data models required for OLAP [CODD93]:

- **Categorical** model: defines the data structures.
- **Exegetical** model: contains the historical data.
- **Contemplative** model: for the exploration of "what-if" scenarios.
- **Formulaic** model: indicates the complex relationship between apparently disconnected variables.

The benefits of a data warehouse include better quality, more cost-effective and quicker decision making, enhanced customer service and possible support for BPR and IS re-engineering. Many articles and books have been written on the subject of data warehousing, and any attempt to summarize them here would be pointless.

Although it is possible to build a data warehouse on a purely physical data element level, most authors make a strong case for building a high-level enterprise data model before attempting to implement a data warehouse.

> "I can't argue that source to target data migration analysis and design deals with detailed column domain definitions that are not addressed at the conceptual level. However, I think the enterprise data model (EDM) plays a critical role in the planning, design and future success of the enterprise data warehouse. It is precisely because the information sourced to the warehouse is constrained by the type of implementation design issues we wrestle with in column mapping, that we need the EDM. The EDM is a key tool that will help us make sure that the warehouse delivers its promised return on investment." [LONG98]

Various consulting companies have developed generic enterprise data models to "jump-start" the development of the data warehouse. [HEIN01] gives following compelling argument for using what he describes as a template data model.

> "Template data models provide a close approximation of what would be achieved during that lengthy period [of planning a data warehouse] at a small fraction of the cost. Their value must be immediately apparent upon inspection. Template data models are not a 'plan-to-plan' or the starting point

> for some deliverable in the future. They are the deliverable that is intended to jump-start and dramatically shorten the planning, analysis and design phase. It takes a few hours to determine their value. Those few hours can save the project months of work and millions of dollars of costs. Template data models are flexible. They are designed to be modified, extended and integrated with other data models. Applied Data Resource Management (ADRM) often presents an example of an Enterprise Data Model, which is approximately 16-feet in length (paperwise), and incorporates 350 tables and 2,500 columns, with 1,000 pages of documentation. It is a compelling argument to tell senior management "You will never have less than what you see before you. You can accurately predict project, resource and staffing costs by analyzing the areas of the models that need to be extended. Furthermore, it can be installed and available in one hour." Contrast this to an expensive, disruptive information engineering effort, which is frequently controlled by a third party, whose results will not be readily apparent for many months.

The ADRM templates are customized on an industry-specific basis, but each industry template shares a a large number of entities (and attributes) with the others. However, they are not available to the public. Bill Inmon, who runs a similar data warehousing consultancy, has made a sample of his high and intermediate level generic data model available on the Internet (see http://www.billinmon.com) and this will be used in this research.

3.7.2 Business Objects

The adoption of client-server technology and object-orientation, and the development of consistent object interfaces (OMG/CORBA, DCOM) gave a great impetus to the area of business objects. The technology and standards have matured sufficiently for business objects to become a real world technology [SIMS94; TAIL95; PRIN96; GALE96; EELE98]. Vendors are scrambling to develop libraries of business objects with universal or at least industry-specific appeal, and considerable work is also being done within OMG to stimulate this process [OMG97].

The latter defines the business object as follows:

> "A business object is defined as a representation of a thing active in the business domain, including at least its business name and definition, attributes, behavior, relationships, rules, policies and constraints. A business object may represent, for example, a person, place, event, business process or concept. Typical examples of business objects are: employee, product, invoice and payment."

Typically three major types of business objects are distinguished: entity objects, process objects and event objects [SHEL96; OMG97; MARS00]. This OMG-endorsed distinction was subsequently used as the basis for the proposed taxonomy as shown in Table 3-3 [OPEN97].

Some OO CASE tools now come pre-loaded with a library of business objects while in other cases more fully developed libraries can be purchased separately as an add-on. Good examples are the BOMA (Business Object Management Architecture) library developed by SES Software, and this is also the way in which IBM's SanFrancisco and WebSphere frameworks are packaged.

Entity Business Objects	Process Business Objects	Event Business Objects
Resource: tangible, intangible, consumed, retained. **Product**: goods or services. **Party**: people, organization, institution. **Transfer**: exchanges, gifts, purchases, sales. **Agreement**: formal & informal contracts. **Plan**: goals, objectives, standards, specifications. **Location**: areas & points in space.	**Selling**: quotation, order taking, deal negotiation. **Promoting**: marketing, positioning. **Procuring**: buying resources, hiring employees. **Developing**: creating new products/services. **Producing**: manufacturing, performing services. **Delivering**: logistics, materials handling, shipping. **Billing**: collecting payment, extending credit.	**Environment time**: season **Calendar time**: year, month, day. **Clock time**: hour, minute, second. **Business time**: fiscal year, tax period. **Elapsed time**: maturity, expiration, past due. **Bounds**: start, stop. **Occurrence**: interruption, "stuff happens".
Examples: customer, order, product, contract, equipment, capacity, address, vehicle, facility, resource.	**Examples:** (major:) order fulfilment, procurement, production, billing; (sub:) quotation, contracting, vendor certification, invoicing, collection, delivery.	**Examples:** inventory low, tank overpressure, employee absent, approval granted, payment cleared, fiscal year end, loan due, order placed.

Table 3-3: Taxonomy of Business Objects [OPEN97]

A major purpose of the business object vision is to build a system assembled from different generic off-the-shelf business objects from independent vendors adhering to a common standard. In practice, these objects are "embedded in a specific technology" to become business system components. Most of today's enterprise models follow the business object modelling approach, as reflected in, For example, the later versions of the ERP systems and their enterprise modelling component tools (ARIS for SAP R/3, DEM for Baan) [PRIC98b]. The original aim for these ERPs is to package their modules into Business Objects with fully described interfaces, allowing adopting organizations to "plug-and-play" their ERP modules with modules developed in-house or provided by other vendors. This was the motivation for the formation of the Open Applications Group (OAG) which produced the OAG Integration Specification (OAGIS) that achieved little acceptance in the market place.

3.7.3 Enterprise Resource Planning (ERP)

Enterprise-wide information systems, or *ERPs*, show that a major portion of information requirements can be met by means of generic information systems, although their implementation is not necessary a sinecure [DAVE98]. The user requirements are not employed for the conceptual design but rather for the final customization of implemented system. Although some claim to be based on deep conceptual models of the enterprise, most are distillations of industry implementations from which higher-level business object models were developed through a somewhat iterative process [CURR98].

The major players in the ERP market are Baan, Computer Associates, Geac, J.D. Edwards, Oracle, PeopleSoft and SAP, with the latter being the market leader both in terms of large company installations and sales volume [PRIC98b]. In addition, there are a large number of smaller players whose systems are often based on scaled-up versions of their integrated accounting packages.

The only publicized model is the SAP R/3 Reference Model [SCHE98], which is implemented using the ARIS modelling tool [CURR98]. Other ERPs are also based on one or several standard reference models, e.g. Baan's Enterprise Reference Models using the DEM tool, although these models have not been publicized. The SAP R/3 Reference Model is by far the best documented, most resourced and most widely validated publicly available enterprise model and it will feature prominently in this research. In addition, an attempt was made to re-engineer Baan's model from the fairly detailed package description in [PERR98].

3.7.4 Standards

Given the wide variety of reference disciplines and the increasing importance of IT in the world economy, it is not surprising that the number of national and international standards relating to enterprise modelling has increased markedly. It is clear that the standard setting authorities are struggling to keep up with the rapid change in the IT industry. This was clearly demonstrated when, searching for the NIST FIPS 99 Guideline ("A Framework for the Evaluation and Comparison of Software Development Tools"), a notification popped up that this, along with another 32 NIST standards, had been withdrawn "because they are obsolete, or have not been updated to adopt current voluntary industry standards" [NIST97]. The IEEE is more laconic about its withdrawn ("archived") standards, such as IEEE 1348: 'Recommended Practice for the Evaluation and Selection of CASE Tools' & 1061: 'Software Quality Metrics Methodology': "archive standards may have value as historical documents" [IEEE01].

Most of the standards mentioned below, have already been discussed earlier, so this section merely aims to give an overview of the types of standards that are applicable, rather than be a complete listing. The most up-to-date versions of the NIST and IEEE standards are found on their respective websites, which illustrate the unstoppable flood of IT standards.

The largest number of standards concerns software quality. As mentioned earlier, [KIT95] lists twenty important IEEE/ANSI standards specifically governing software quality for the period 1983-1993 over a 10-year period. A more updated version, listing the more recent ISO/IEC/IEEE software quality standards can be found in [WALL01, p. 11-16]., which covers the more important standards up to 2000. A much more comprehensive survey covering the standards pertaining to the entire software engineering field is the somewhat dated Magee & Tripp Annotated Directory of Standards and Specifications [MAGE94] which lists hundreds of SE standards originating from 44 different standards issuing organizations. To give a feeling for the range and diversity of the standards, here are some of the more "interesting" or relevant ones in "Organization - Standard Code (Year): Name (number of pages)" format:

- CEN - ENV 40 003 (1990): CIM Systems Architecture Framework for Modelling (26 p.).
- ECMA - TR/55 (1991): Reference Model for Frameworks of Software Engineering Environments (97 p.).
- EIA - IS-81 (1991): CDIF - Framework for Modeling and Extensibility (101 p.).
- IEE - SE-4 (1990): Software Quality Assurance Model Procedures (107 p.).
- IEEE - 610.12 (1990): Glossary of Software Engineering Terminology (83 p.).
- IEEE - 830 (1993): Software Requirements Specifications (tentative) (28 p.).
- IEEE - 982.1 (1988): Dictionary of Measures to Produce Reliable Software (36 p.).
- IEEE - 982.2 (1988): Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software (96 p.).
- IEEE - 1002 (1987): Taxonomy for Software Engineering Standards (20 p.).
- ISO - 2382 (1984-92): Information Technology - Vocabulary (1000+ p.) e.g. ISO 2382-20 is part 20 containing the terms related to system development.

- NISO / ANSI - Z.39.67 (1991): Computer Software Description (33 p.).
- NIST FIPS PUB 11-3 (1991) (the old ANSI X3.172): American National Dictionary for Information Processing Systems (150 p.).
- ANS / ANSI - 10.4 (1987): Guidelines for the Verification and Validation of Scientific and Engineering Computer Programs for the Nuclear Industry (36 p.).
- ANS / ANSI - 10 (1986): Guidelines for Considering User Needs in Computer Program Development (6 p.).
- ANS - X3.88 (1981): American National Standard for Computer Program Abstracts (6 p.).
- ASTM - E.31 (1990): Guide for Selection and Acquisition of Commercially Available Computer Systems (7 p.).
- BSI - BS 6154 (1981): Method of Defining Syntactic Metalanguage (14 p.).
- IEEE - 1059 (1993): Guide for Software Verification and Validation Plans (20 p.).
- IEEE - 730 (1987): Standard for Software Design Descriptions - now superseded by 730-1998.

Sheppard mentions the following "9001-type" overall software quality standards [SHEP95]:

- ISO 8402: Definition of quality vocabulary
- ISO 9001: International quality management and quality assurance standard for manufacturing processes.
- ISO 9000-3: The interpretation of ISO 9001 with regard to the software industry.
- BS 5750 Part 1: British standard equivalent to ISO 9001.
- EN 29001: European standard equivalent to ISO 9001.

Since then, the ISO/IEC released their standard 9126 in 1998 on "Information Technology-Software Product Quality", which focuses on a proposed definition and description of six broadly defined and described quality characteristic relating mainly to finished software products: functionality, reliability, usability, efficiency, maintainability and portability. Other related standards are [WALL01]:

- IEEE 929 (1998): Software Test Documentation.
- IEEE 982.1 and 982.2: Standard Dictionary of Measures to Produce Reliable Software.
- IEEE 1012 (1998): Software Verification and Validation.
- IEEE 1028 (1997): Software Reviews.
- ISO/IEC 14598 (1998): Software Product Evaluation.
- ISO/IEC 25939 (2000) Information Technology – Software Measurement Process.

More directly related to enterprise modelling is the approach taken by [NELL99] who concluded that the best area for standards in enterprise modelling concerns the design of interfaces. Some of the data interchange formats relevant in this regard are the ISO 10303, STEP, EDIFACT and ANSI X12.

The interchange between modelling tools was governed by CDIF although it was criticized by [SARR96] as being too voluminous and unwieldy. There are two standards concerned with the actual data dictionary / repository field. One is ISO/IEC IS 100027: IRDS (Information Resource Dictionary System), with a 4-layered architecture similar to CDIF's and MOF's, but it seems to have disappeared into obscurity. The ECMA/ISO/IEC DIS 13719 PCTE (Portable Common Tool Environment) has similarly been overtaken by the move towards object-driven standards: the CORBA interoperability and CDIF-inspired MOF "semantic bus" proposal by OMG [BEZI98a], ODMG's ODL (Object Definition Language), or the Metadata Coalition's MDIS (MetaData Interchange Specification). ISO/IEC's RM-ODP (Reference Model of Open Distributed Processing), a meta-standard for the specification of open distributed systems, never seems to have had any market clout, since most vendors were included in the OMG. But even object-driven standards will probably be superseded by an XML-based standard such as XMI.

The various ontology exchange standards and markup languages, useful not only for the exchange of ontologies but also knowledge, were discussed above. We also saw the shift from the AI logic-based

languages such as KIF, FOL, KQML, CML etc. to the XML-based SHOE XOL, OML, CKML, RDFS and RDF-based OIL and DAML+OIL.

A critical problem in modelling is that of a consistent terminology. Standardization proposals in respect of terminology are the six-part ISO/IEC 11179 Data Element Standard [GILL97a], the X12 / EDIFACT-driven BSR (Basic Semantics Register) and the Universal Data Element Framework (UDEF) which is analogous to the Dewey decimal library cataloguing system. All three incorporate a registration procedure for the continuous addition and updating of new data names, with the ISO/IEC 11179 registry method extended in ANSI X3.285. UDEF, following the approach of many ontology-based proposals, does not require that one adopts the names provided, but merely that different modellers map their terminology to the UDEF name base. The 8320.1 data elements standard, established by the US Department of Defense (DoD), is roughly analogous to the ISO 11179. While the DoD has managed to populate its registry with over 17 000 standard data elements, the quality of its registry is debatable since there appears to be a great deal of redundancy and poor element design, in addition to the very complex naming structure which leads to sometimes unusable names [ROSE97; HAYE99].

The cumbersome and slow-moving EDIFACT standard never obtained the market acceptance that was anticipated, except for some large corporates and a number of vertical industry specializations. It has received some competition from the rather still-born ERP-driven OAGIS group. Both are likely to be overtaken by an XML-based alternative, although these alternatives are going to have to let the market decide the battle: some early contenders are Microsoft's BizTalk, Veosystem's Common Business Library (CBL), RossettaNet and Ariba's Commerce XML (cXML) [FENS01b]. It is possible that, here again, smaller and more agile standards will first develop for specific vertical industries such as ICE (Information and Content Exchange Protocol) for content syndicators; IOTP (Internet Open Trading Protocol) for electronic payments; OAGIS (mentioned above) for ERP-to-ERP data transfer; or OFX (Open Financial Exchange) for financial institutions.

Enterprise engineering also produced a number of standards, driven mainly by the TC184 SC5 WG1 workgroup on industrial-automation systems and integration. A first standard, discussed above, was the ISO 14258: *concepts and rules for enterprise models* [NELL96; NELL99]. ISO 15704: *requirements for enterprise reference architectures and methodologies*, followed soon after. The latter specifies the criteria to which a reference architecture must adhere order for it to be considered "complete", and is based in part on the old CEN ENV 40003 [KOSA99].

3.8 Management Sciences

Management sciences are intimately concerned with the enterprise. The following disciplines contribute towards generic enterprise modelling.

3.8.1 Accounting

The accounting model is the oldest known general enterprise model to serve as the basis of a business information system. In fact, the oldest survived writings, Mesopotamian clay tablets, are assumed to be records of business transactions. The *accounting model* in its current form can be traced back to Luca Pacioli's introduction of the double entry accounting system in 1494[2]. Despite, or perhaps

[2] Although he was not necessarily the inventor of double entry accounting, his "mathematics" text, which included a comprehensive section on the accounting system, was the first time all new accounting innovations had been combined in a single volume. From there, the "Italian way" of accounting spread quickly throughout Europe.

because of, its simplicity the accounting model has formed the basis for the great majority of business information systems: its model underlies most of today's transaction processing and management information systems.

Although the model was created five centuries ago, it already incorporated many critical IS components: a taxonomy of entities (with semantic interpretations), business logic rules, a number of attributes or data dimensions, distinction between the TPS and MIS components, and control/audit procedures. In spite of its shortcomings (historical view, essentially two-dimensional, ignores many relevant entities and events etc.) it is a cornerstone of virtually all transaction processing systems. The accounting model was initially developed for "for-profit businesses" although it is currently used for any type of organization and even by individuals.

In a way, the general ledger accounting database can be considered to be a data warehouse with five dimensions [GLAS01]:

- time (fiscal period and year),
- organizational entity (business unit),
- natural account (object account),
- sub-account (subsidiary account), and
- ledger type (e.g. actual, budget, statistical).

Although the basic model is used globally, the national accounting bodies in most countries have adopted and/or expanded the model in specific ways. Usually this is in the form of specific government legislation, or through the publication of standards and guidelines by the national accounting body. Most of the modifications are more detailed models or guidelines on the application and interpretation of the accounting rules.

The most detailed models have been developed in Western Europe, perhaps because of an arguably more pronounced culture of statutory public accountability. This is evident in the 4th directive promulgated in 1972 by what was then the European Economic Community, aimed at the harmonization of accounting procedures and financial statements. Although the directive was based in part on the German practices, Belgium was the first country to translate the directive into specific legislation, and it provides an excellent example of how a fairly detailed standard has been applied across industries[3]. The "Belgian Royal Decree on Accounting Standard and Financial Reporting" (1983) prescribes the accounting rules and a chart of accounts to a very detailed level – several hundreds of accounts are prescribed and need to be reported by all medium-sized and large businesses. [SHAR99]

Many organizations are subject to legislation that prescribes public accountability in the form of periodical financial statements with defined accounts (and reporting rules). For instance, a publicly listed company may be required to publish different statements to its shareholders and tax authorities. The situation is compounded for multinational organizations that need to conform to standards from different nations which may conflict. In the worst case, the differences in requirements may force an organization to operate a number of different sets of books: one for internal purposes, one for income tax purposes, one for international headquarters etc. even though the majority of transactions will not be affected. In what follows, the focus will be on the external reporting requirements, which is the area most commonly regulated.

[3] All larger business enterprises have to submit their annual statements to a centralised national databank that acts as a public clearing house of corporate financial information.

The accounting model provides in principle for an unlimited number of entities as reflected in the *chart of accounts*. Organizations are expected or assumed to extend the chart of accounts to cover *all* the entities for which it wishes to record information: each entity of interest is represented by means of an account. Many larger organizations have indeed many thousands of accounts.

A chart of accounts is structured in a hierarchical fashion, which can be translated into a taxonomy where the highest levels are fairly standardized but the lower levels tend to be industry- and (even lower) organization-specific. Although internal accounts can be set up in an arbitrary fashion, there are usually specific requirements for external reporting.

There is general consensus that substantial areas of business information are left out in the traditional accounting model. For example, in the assets arena alone, the following critical assets are generally not "accounted" for: human resources, assets under control but not owned, long-term sale contracts and agreements, goodwill, organizational knowledge.

The basic accounting model takes a very restricted view of the organization. The following are just some of the limitations:

- **Historical perspective**. Only business events that actually happened are recorded. The standard model does not include forecasts (e.g. budgets). In addition, there is often a delay between the event and the reflection of that event in the financial statements. This lag often infuriates business decision-makers.
- **Monetary perspective**. No explicit support for natural or physical units is provided. All events have to be translated into monetary terms, resulting in many problems relating to valuation models and historical changes in purchasing power of the monetary unit. Even the monetary perspective is limited: it is a (conservative) point value leaving no scope for probabilities, risk profiles, rounding effects.
- **Scope**. A large number of significant business events are not recorded in the accounting model, for various reasons. Some entities are considered not to be material enough, such as human resources, organizational knowledge or information assets (customer database, system development). Some attributes of even the "material events" are not captured: the scenario of which it is part (e.g. a long-term sales contract entails a regular stream of sales; an investment in a fixed asset implies a number of depreciations and final asset disposition), the time and location, actor and decision maker aspects, physical units, reliability / probability indicators. Finally, a number of events are not recognized, for example moving a fixed asset from within one building to another.

The accounting model provides the architecture for most of the transaction processing systems that are developed under the paradigm of functionally oriented systems i.e. the "data processing" approach to information systems with the focus on the recording of business transactions, although many of these systems have fairly elaborate management reporting facilities.

Interestingly [JOHN01] explores the use of the accounting model as a generic paradigm basis on which to develop more general business transaction processing systems.

3.8.2 Finance

Financial managers of most large organizations who have to handle large financial budgets use *financial models*. The larger the organization (and its budget), the more formal and elaborate its model tends to be. Although the accounting model is the financial managers' framework of reference, its focus is the recording of historical business events. The accounting model does not support forward-

looking decision-making processes such as planning, forecasting, long-term budgeting, simulation, what-if analysis, probabilities and scenarios. Financial business models were developed to facilitate the latter type of analysis. Unlike the accounting model, there is no statutory or de-facto standard financial model. Most organizations have developed or customized their own models though there tends to be a significant overlap between the models.

Financial models range from simplistic user-developed once-off financial spreadsheets to extremely sophisticated mainframe software produced by the likes of IBM. Many of these financial models are really extensions of the accounting model into the realm of decision support systems. As an illustrative publicly available example, the somewhat dated but still valid USB Growth Model will be used in this research [VANB88].

There is a substantial body of literature in the fields of decision support systems, spreadsheet modelling and the budgeting modules of ERP and TPS that relates to financial models.

3.8.3 Business Science, Organizational Theory and Economics

The field of **business management** has the enterprise as its main object of interest. Hence the model of the generic enterprise is an important construct. A multitude of abstract business models and simulation models, including linear programming and other optimization models, have been studied in the field of operations research [STER00]. In addition, a lot of business modelling techniques were developed with BPR in mind. However, not many theoretical contributions towards building information systems models were made. Rather, the reference disciplines mentioned earlier contributed towards the development of IS which included DSS, MIS and EIS components.

The field of **micro-economics** also takes the enterprise as its research object. However, the models built in micro-economics tend to be mathematical constructs supporting the evaluation of theories and hence abstract models, rather than usable for modelling real-world practical information systems.

Organizational theory also studies organizations, but does not look at the individual enterprise. It takes an inter-organizational perspective e.g. how organizations are structured, how they grow, the role of corporate culture and value systems [DAWS96]. Typically organizational theory will compare organizations in time or across different cultures. As such, the discipline provides many penetrating analyses and useful concepts, but no concrete "enterprise model" which can be used in the context of this research.

3.9 Other Related Disciplines

There are many other disciplines that have made contributions to the field of enterprise modelling.

Linguistics provides important methods for text analysis which will be used in the validation of models, such as keyword in context, similarity analysis [HONK98] and semantic analysis. From **semiotics**, the study of signs, came the important distinction between syntax, semantics & pragmatics of language, concepts equally applicable to information models [COLO95]. Goosenaerts proposes the term *industrial simiosis* to indicate the study of the process by which knowledge is formed and applied during industrial processes and professional work i.e. the discipline "concerned with the interplay of signs, data, models and physical objects, both artefacts or products, and machines or resources and interpretants, applied by people and information systems during their work" [GOOS00]. [STAM87] used linguistics as a point of departure to pose a number of questions about the impact of IS data models on semantics, mostly related to the diversity of meanings of words and data elements according to different system users. Stamper argues strongly *against* the use of formal techniques and a platonic realism stance in the context of organizations.

Applied mathematics also provides some valuable analysis techniques which can be applied to model analysis. Apart from the standard statistical techniques, these include formal concept analysis, complexity measures, and graph analysis. Each of these are worthy of their own chapter but will be introduced in more detail in the analysis section.

This overview of reference disciplines concludes the literature survey. The next chapter will specify the methodology adopted for the research.

Chapter 4: Methodology

Having investigated the nature of modelling and the various reference disciplines that engage in enterprise modelling, it is now possible to discuss the methodology that will be adopted in this thesis. Note that the term 'methodology' here is used in the sense of research methodology; not to be confused with the sense of IS methodology (refer e.g. IS method engineering) as used in most other sections of the thesis.

4.1 Overall Methodology

This research is aimed at developing and validating a comprehensive framework for the analysis and evaluation of enterprise models. Gorman presents an informal workplan for evaluating data models in [GORM98] but, apart from its simplicity and lack of theoretical foundation, it is not very applicable for models from different reference disciplines.

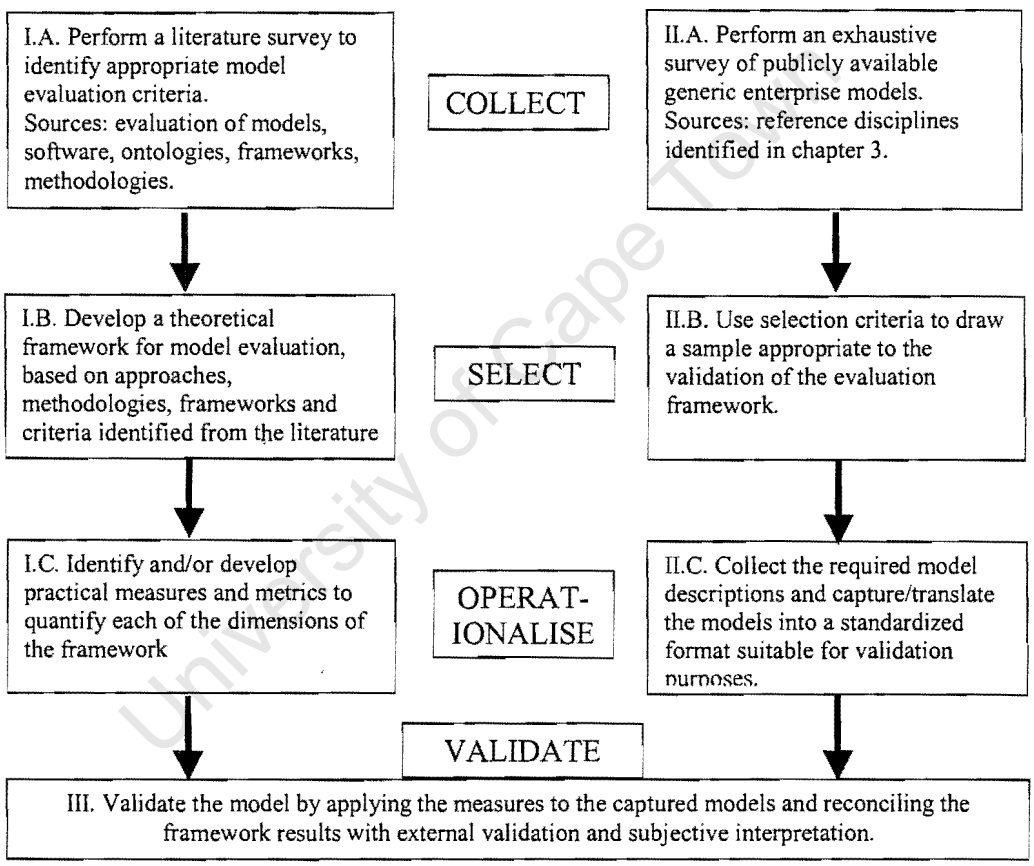


Figure 4-1: Overview of Methodology Used in this Thesis.

The methodology to achieve this consists of a number of steps as shown in Figure 4-1. This methodology is a substantial refinement of what was originally presented in [VANB96].

Steps I.A to I.C are concerned with the development of a comprehensive framework. To this end, a wide range of criteria, which could be applied to model evaluation, will be identified from the literature. Most of these will be concerned with the evaluation of other but related "intellectual products" such as packaged software, ontologies, or methodologies. These will be used as the basis from which to build the framework. Note that although a number of independent measures and metrics for model evaluation exist, these typically stand on their own. The proposed framework will integrate these measures by adopting a number of different perspectives and relating these criteria to

each other. Finally, it is not sufficient merely to develop a framework with theoretical criteria; these criteria need to be operationalized into practical measurements and/or metrics to enable the objective applicability of the framework. Steps I.A and I.B will be done in Chapter 5 and step I.C in the analysis chapters (7 onwards). Some of the metrics introduced are new contributions and should be of interest to IS modellers and researchers alike. It is also important to note that the framework should be adaptable with minor substitutions to other domain areas i.e. models that do not model the enterprise.

The methodology proposed for the construction of the framework is, in fact, very congruent with the Common-KADS methodology for the construction of a knowledge model, which consists of the following three main stages (with a sample of typical activities) [SCHR98; MEIS95]:

- Stage 1: Knowledge Identification. Explore information sources and list potential components for reuse.
- Stage 2: Knowledge Specification. Construct initial domain conceptualization and complete the knowledge-model specification.
- Stage 3: Knowledge Refinement. Validate knowledge model.

By substituting "knowledge model" with "framework", one arrives at a very similar approach to the one described in steps I.A to I.C.

The development of a theoretical framework does not contribute towards science unless it is accompanied by a proper validation of the framework itself. This involves creating a sample of enterprise models and applying the framework to the sample. A substantial number of enterprise models from different reference disciplines (as discussed in Chapter 3) and from both commercial and academic origins will be surveyed in Chapter 6 (step II.A). This chapter will look at the minimum criteria necessary for forming an appropriate sample (step II.B). Summary descriptions of the selected models will be available in Chapter 6 whilst most of the technical issues relating to the model capture and conversion (step II.C) will be documented in addenda. The process of capture, coding, integration and validation of the sample models is loosely based on the process for the development of ontologies as suggested in [USCH96b].

The database containing these enterprise models in a common format (XML) will then be used in the analysis chapters (7 onwards) to validate the model. Both the database itself, as well as the analysis generated from the framework, should form a noteworthy contribution for IS practitioners.

4.2 Framework validation

Scientific progress should not be measured by the development of a large quantity of frameworks and theories, but rather by the quality and originality of these theoretical and explanatory constructs. Popper and Kuhn's writings have caused the philosophy of science to put even greater emphasis on the necessity of empirical validation. The exact methodology for this validation process, however, is less clear and depends on the discipline as well as on the scope and intentions of the theoretical contributions.

The concept of *validity* is normally applied to mathematical models and theories in the context of statistical and experimental design. This statistical interpretation of the concept of validity will be explored in more detail in 4.2.3 below.

At a more conceptual level, Preece suggests five possible validation approaches for knowledge-based systems [PREE98]:

- Inspection by human proof-reading, the most commonly used method.

- Static verification: checking for logical anomalies i.e. correctness.
- Formal proof: a rare method depending on a very formal modelling language but not applicable to the enterprise domain.
- Cross-reference verification: checking the internal consistency.
- Empirical testing: by checking the implemented system against test cases.

These approaches can also be used or applied to the validation of a framework. The inspection and static verification approach have obviously been applied in great detail by the author and researchers who have reviewed earlier drafts of the framework. In fact, this has contributed to the abandoning of an early incarnation of the framework as detailed in 5.2.8. Additional inspection validation will result from the examining process to which this research is subject, as well as subsequent publication of this research.

The formal proof method is unfortunately not applicable since the language used in formulating the framework and specifying the applicable domain is not suitably formalized. The main methodological approach is therefore inductive.

For the purposes of this research, the validation of the framework shall therefore take two forms: the internal i.e. theoretical or conceptual validation, which subsumes the cross-reference verification, and the external i.e. empirical or operational validation.

4.2.1 Theoretical or Conceptual Validation

The theoretical validation of the framework is concerned with its foundations and internal structure, without reference to the actual usability. Although there is no definite list of criteria, some commonly used principles for frameworks are self-consistency, face validity, elegance, generality, efficiency (Occam's razor), comprehensibility etc. Chapter 5 conducts an exhaustive search of criteria for good and methodologies and this will not only inform the process of populating the framework with suitable criteria for model evaluation, but it also provides criteria for evaluating the framework itself. This will allow an initial conceptual framework validation in section 5.4.

The framework can, in a *tour-de-force*, also be used in a self-referential manner to validate itself. Although this process may seem somewhat suspect from a methodological point of view, this approach has been demonstrated before: the suggestion that the Zachman framework can be applied to itself or, more pragmatically, using UML notation in the meta-level definition of the UML [OMG99]. This involves a reorganisation of the material presented in section 5.4 and will be presented in section 10.3 in the context of the generalisability of the framework.

4.2.2 Empirical or Operational Validation

Notwithstanding the above, it is interesting to note that the field of systems engineering actually has an official definition for validation. The following is the IEEE/ANSI definition of validation: "Validation is the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements." [KIT95, p.28]

The approach taken by the IEEE is a highly practical one: how well does the system work in practice. Although the definition is intended for software systems, it applies equally well to the validation of the proposed framework: how well does the proposed framework achieve its stated purpose of evaluating and comparing enterprise models in practice? This is arguably the most important form of framework validation since it addresses the practical usefulness and applicability of the framework. Hence this will take up the bulk of this research and is discussed in chapters 7 to 9.

It is suggested that a high confidence in the operational validity is ensured by a the comprehensiveness of the sample of enterprise models, although the sample is possibly biased by the non-inclusion of the commercial "off-the-shelf" enterprise models, which one might expect to be of higher quality. Some suppliers of these models were asked to donate an inspection copy of their model for research purposes but this approach was unsuccessful. A more detailed discussion of the sample validity is found in 4.3 below. However, the application of the framework and its criteria to real world models introduces an extra validation concern.

4.2.3 Criteria for Evaluating and Validating the Framework Metrics and Measures

The operationalization of the framework introduces a new validity issue: for many of the constructs contained with the framework, new metrics or measurements will be developed. For others, existing measures will be used. However, these metrics themselves need to be validated insofar that they are empirical instruments aiming to measure abstract constructs (namely the framework criteria). Therefore, the discussion and interpretation of each of the framework criteria needs to address both the validity of the metric or measuring approach used (i.e. as a measuring instrument for the criterion) as well as evaluating the criterion in the context of the sample.

Unlike for the case of a theoretical framework, there exists a strong body of statistical literature on how to assess the validity of empirical instruments. In fact, a large number of distinct types of validity have been proposed, each looking at a different dimension or aspect. [LEED01] mentions six types of validity. These will be used as the evaluation criteria for the metrics investigated in operationalizing the framework in the chapters 7 to 10. The list below includes a description along with a brief discussion on how they can be applied to the metrics and measures to be suggested in this thesis.

- **Face validity**: a subjective judgement of whether the metric measures what it is supposed to measure, whether it is sufficiently comprehensive and whether the validation sample is representative. This type of validity will generally *not* be mentioned in the evaluation of the metrics because metrics that do not have sufficient face validity would not be considered for this research study in the first place. It can therefore be assumed that all metrics have a fair to high degree of face validity.
- **Criterion validity**: are there standards against which to check the results of the individual measures, and how well are these standards actually measured in the framework? Most of the measures to be developed have no objective or intrinsic value: many of the values are of a comparative nature allowing a ranked comparison rather than an absolute rating. For a few selected metrics, there exist theoretical minimum and/or maximum values (e.g. model complexity) but these carry little practical importance. It is intended that this research, by means of the large sample, will set benchmark values for the various criteria so that these values can be used by future researchers or practitioners adopting the framework. In addition, so-called "random models" with varying degrees of semantic randomness have been created to set base-line values for some measures.
- **Content validity**: the accuracy with which an instrument measures the factors under study. One of the prime purposes of the framework is to increase the content validity of a number of earlier proposed model measures. This will be done by offering quantitative and operational measures e.g. to measure and compare model structure, complexity, family likeness, style or quality. Naturally, this will be a first (though well-reasoned and defended) iteration; and future research may well, and should, improve on the proposals made in this thesis.

- **Construct validity**: how well does the framework measure or approximate theoretical constructs which cannot directly be observed or isolated? In an effort to enhance construct validity, two approaches will be taken. The first is to use a number of alternatives, i.e. different, metrics or instruments to measure each of the framework constructs and to compare their results for each of the models. Rankings that change dramatically depending on which metric is used may be indicative of low construct validity. The second approach is to do sensitivity analysis on factor weightings i.e. by changing the parameters of composite indices and measures and checking how this influences the ranking or score. Again, high (rank order) sensitivity will be indicative of low construct validity.
- **Internal validity**: the freedom from bias i.e. informing conclusions in view of the data. Does the manner in which the framework is adopted influence the final results? Since the framework will not include a process specification, it is suggested that the framework will have a high internal validity. All measures used in the operationalization of the framework will be completely objective and repeatable. Unfortunately, a number of decisions will need to be taken in the selection and capture of the models, which introduces an element of bias. Such decisions will be therefore be fully documented and motivated.
- **External validity**: how generalisable are the results i.e. can one extrapolate the findings from the sample to the universe? For this research, the universe is defined as "enterprise models". Because the sample will comprise a significant portion of universe (i.e. all available models), and stretches across a number of different disciplines and modelling paradigms, the external validity is believed to be very high. The two main reservations concern the lack of commercial "pure" models (see note above) and the restriction to structural models (see below). On the other hand, it is strongly believed that the framework can easily be adapted or generalized by extending the universe to include other modelling domains e.g. certain functional areas within enterprises or more technical domains. Section 10.3 will go one step further and suggest that the *framework* can also be applied in non-modelling contexts, but this does obviously not extend to claiming that the same measures or metrics are valid in the new context. A separate validation process would be necessary as suggested in 11.6.4.

A more structured and comprehensive listing of the different types of statistical validity is given by Sekaran [SEKA03]. This includes all the types of validity given above but adds some more detailed (sub-)types).

However, this larger set includes more refined types of validity that are much more amenable to statistical analysis involved huge samples than to the metrics and sample used in this research.

4.3 Generating the Sample of Enterprise Models

The selection of models for evaluating the framework needs to be guided by certain criteria. Unfortunately, the normal statistical criteria to select an unbiased sample from a population do not apply, since the population is too small. In fact, the sample is intended to be as large a subset of the population as practically possible. In addition, the purpose is not to obtain an unbiased or representative sample, but rather a sample that will best validate the framework i.e. spread and heterogeneity are more important than representativeness. Unfortunately, the scientific literature is quiet on the subject of how to select a sample to validate a theoretical framework; hence the following is based on personal opinion.

Table 4-1: Sekaran's Types of Validity.

Type of Validity	Sekaran's description [SEKA03:206-208]
1.1 Internal validity	Are the cause-and-effect relationships authentic, meaningful and useful?
1.2 External validity	Is the model generalisable to the external environment?
2 Content validity	Does the measure adequately measure the concept?
2.1 Face validity	A way of measuring content validity: do "experts" validate that the instrument measures what its name suggests?
2.2 Authoritative validity	
3 Criterion-related validity	Does the measure differentiate in a manner that helps to predict a criterion variable?
3.1 Concurrent validity	Does the measure differentiate in a manner that helps to predict a criterion variable currently?
3.2 Predictive validity	Does the measure differentiate in a manner that helps to predict a future criterion?
4 Construct validity	Does the instrument tap the concept as theorized?
4.1 Convergent validity	Do two instruments measuring the concept correlate highly?
4.2 Discriminant validity	Does the measure have a low correlation with a variable that is supposed to be unrelated to this variable?

An instinctive reaction is to select "quality" models, i.e. using general theoretical criteria for a "good scientific model" e.g. Occam's razor, universality, expressiveness, logical completeness, consistent notation. [BENY90] lists the following pragmatic criteria for good models: aids clear thinking, readable by end-users, graphical manipulation, good basis for communication. Fox, Chionglo and Fadel [FOX93a], in their search for a common-sense model of the enterprise, suggest more specific criteria: generality, competence, efficiency, perspicuity, transformability, extensibility, granularity and scalability. However, selecting only high-quality models would bias the sample since the framework is exactly intended to measure, *inter alia*, the intrinsic quality of the models. Hence quality should not be used as a selection criterion, although high quality models should be incorporated.

4.3.1 Proposed Selection Criteria

The following, then, are the proposed criteria for model selection from the "universe of models":

- **Inclusivity**. The aim is to include as many models in the sample as possible. No model that meets the other criteria will be excluded.
- **Minimum size**. To avoid the inclusion of simplistic models, a reasonable minimum number of entities and relationships is required. A model consisting of, say, 10 entities and 15 relationships is too simplistic to be considered in terms of the objectives of the framework. Although there is no theoretical reason to adopt specific numbers, an enterprise model should have at least 50 to 100 entities and about twice as many relationships.
- **Public availability**. To ensure replicability of the research, the models must be fairly easily accessible to other academic researchers. This does not mean that the models must be available in the public domain (few if any of them are) but that the source materials documenting the models must be available in research publications, books or on the Internet. A particular dilemma is the availability of "commercial models"; these are available but typically at significant charge (e.g.

$100,000 for the Universal Data Model). These models will be mentioned in the survey but since the high cost makes them out of reach for many researchers, they will *not* be considered as publicly available.

- **Reference disciplines**. A deliberate effort has been made to include models from as many disparate disciplines as possible, in order to stress-test the limits of the framework.
- **Quality**. As stated above, a number of models whose quality is apparently lower than that of others will be included since quality is exactly one of the dimensions that the framework is assessing.
- **Modelling Paradigm/Notation**. An endeavour has been made to obtain a spread in modelling paradigms: the sample is to include models from different notational/paradigmatic backgrounds: OO, EER, ER, Knowledge-based and natural language.

It is also important to note that the data for the selected models is identified from the published description as per cited materials. In a number of cases, there were some discrepancies between e.g. the model as imbedded in some source code and the model description in the accompanying documentation. Since the source code may include technical implementation issues and/or coding errors, the printed documentation was taken as the primary source. While this may introduce some error, this is likely to be fairly small and should be attributed to failure of the model documenters in making their full specifications explicit. Note that this is the same approach as adopted by [JAYA94] in his methodology evaluation and by NCITS in their evaluation of object models:

> "It is important to realize that the object model described here is not necessarily the object model of the author(s); it is the result of an attempt to capture the object model described in the book. The book may have been misinterpreted; the book may have used only part of the author(s) complete object model; or that model may have changed since the publication of that book." [NCIT97:4]

4.3.2 Data or structural models.

In principle the framework should be formulated and validated in the context of any possible enterprise model. However, the validation of the framework will be done against a sample of *static* enterprise models for the following reasons:

- Most enterprise models are available only in "data model" format; there are far fewer generic process models than data models. Even most generic OO enterprise models emphasize the structural elements and consist primarily of static class diagrams.
- There is much less variety across enterprises in the data structure than in their processes. This makes comparison easier (more similarity) and also accounts for the lack of generic process models, which tend to be more industry-specific.
- Data structure is more stable than processes [LOCH98].
- There is more uniformity in the modelling methodologies for data models than dynamic models.
- A number of important applications of models are much more amenable to data models e.g. data warehousing, EDI or XML.

In conclusion, the sample will be restricted to static models i.e. ER, EER, OO class diagrams, and their equivalent in the other reference disciplines. In the development of the framework, extensions will be suggested to cater for dynamic models, although it is not suggested that this is a trivial extension.

4.3.3 Random Models to Serve a Base-line Models

In order to assess quality, style and other intrinsic model attributes, it is necessary to have a base line or reference. This will be done by means of the inclusion into the sample of a "randomly generated enterprise models", perhaps better described as *quasi-models*, i.e. models that look like real models but are actually the result of a random generation process. Four random models were created, listed below in increasing order of semantic richness but with an attempt to retain as much as possible an identical syntactic structure.

4.3.4 Random

This is intended to be a fully random model i.e. a model without semantic content, containing 258 randomly selected English words (the concepts/entities) and 455 relationships between them. (The numbers are designed to match the numbers of the semantically richest "OttawaDense" model below). It is designed in such a way that each concept contains at least one link. This model is called "Random1". The words were selected from the Oxford Paperback Dictionary [OXFO79] by selecting the column head words on approximately each third page. Where the headword was not a noun (or where it was a proper noun or double word), the next column would be selected. Because the dictionary is just short of 800 pages, the necessary extra nouns were added at appropriate relative intervals by skipping just two pages instead of three. 455 pairs of random numbers were generated. These accounted for the 455 random relationships, with each pair of numbers corresponding with a "from" and a "to" entity. To create random generalizations (sub/supertype relationships), the average ratio of generalization relationships to concepts/entities in all the captured models (generic model database) was calculated. This ratio, 70.1%, was applied to the number of entities in the random model, to yield a desired number of 320. In fact, 322 random generalization relationships had to be created, since 2 were "reflexive" i.e. entity numbers 168 & 246 had themselves as their supertype!

Note that the random model could have been made even more random by using nonsensical random character strings instead of existing English words. There may be a bias in the headword selection since words with multiple meanings are more likely to be included than single meaning terms (the former occupy more dictionary space). This should not impact the analysis.

4.3.5 Semi-Random

This is intended to be a semi-random model, with entities related to the domain (the organization) but with meaningless relationships between the entities. It contains the 258 English business words selected from the OttawaDense model below, but with the same 455 random relations between them as in the Random model. The same inheritance structure as for the random model was retained.

4.3.6 OttawaBig and OttawaDense

This is a semantically based "hyperlink" version of a business terms dictionary i.e. a "model" consisting of domain-specific concepts with relationships representing semantic references in the lexicon definitions of the concepts.

The source lexicon was the "Business Dictionary" available on the website of the Ottawa Business Journal as available on 17 November 2001. It has just over 900 entries with definitions, frequently containing references to other terms in the dictionary, indicated by capitalization. These references can be thought of as hyperlinks though they are not implemented as such in the online source lexicon. The following is a sample definition:

RECAPITALIZATION:
Alteration of a corporation's CAPITAL STRUCTURE, such as an exchange of bonds for stock. BANKRUPTCY is a common reason for recapitalization; debentures might be exchanged for REORGANIZATION BONDS that pay interest only when earned. See also DEFEASANCE.

This would be interpreted as the entity or concept "RECAPITALIZATION" with 4 relationships from this to the concepts of CAPITAL STRUCTURE, BANKRUPTCY, REORGANIZATION BONDS and DEFEASANCE respectively. In this case, the term REORGANIZATION BONDS turns out not to be a head entry in the lexicon; both the individual terms of REORGANIZATION and BOND actually *are* headwords but mean something else, so only three relationships would actually be recorded in this case.

After deleting a few spurious key terms (e.g. R&D: See RESEARCH AND DEVELOPMENT) and correcting some of the spelling mistakes, 901 terms with definitions were retained. In these definitions, a total of 1050 references was made, of which only 634 could be identified as referring to one of the 901 terms. The relationships linked only a subset of 459 terms, implying 442 "orphan" entities i.e. terms that do not refer to, and are not referred to by, any other terms in the lexicon.

The 459 entities, together with the 634 relationships linking them, form the "**OttawaBig**" model. Of these terms, 201 terms participate in only one single relationship. In order to create a more compact model with a denser network of relationships, these terms with the relationship linking them were deleted to form a second model, namely the "**OttawaDense**" model consisting of 248 entities and 455 relationships. Some relationships linked 2 "single-relationship" terms, so less than 201 relationships were removed from OttawaBig.

4.4 The Basic Meta-Model for Sample Model Capture

Since the enterprise data models in the sample originate from different methodological backgrounds and vary in the amount of detail, there is a need to standardize the type of data captured.

Figure 4-2 illustrates the meta-model for the model database. The meta-model is important in that it determines the exact type and the nature of the data which has been captured for each of the models.

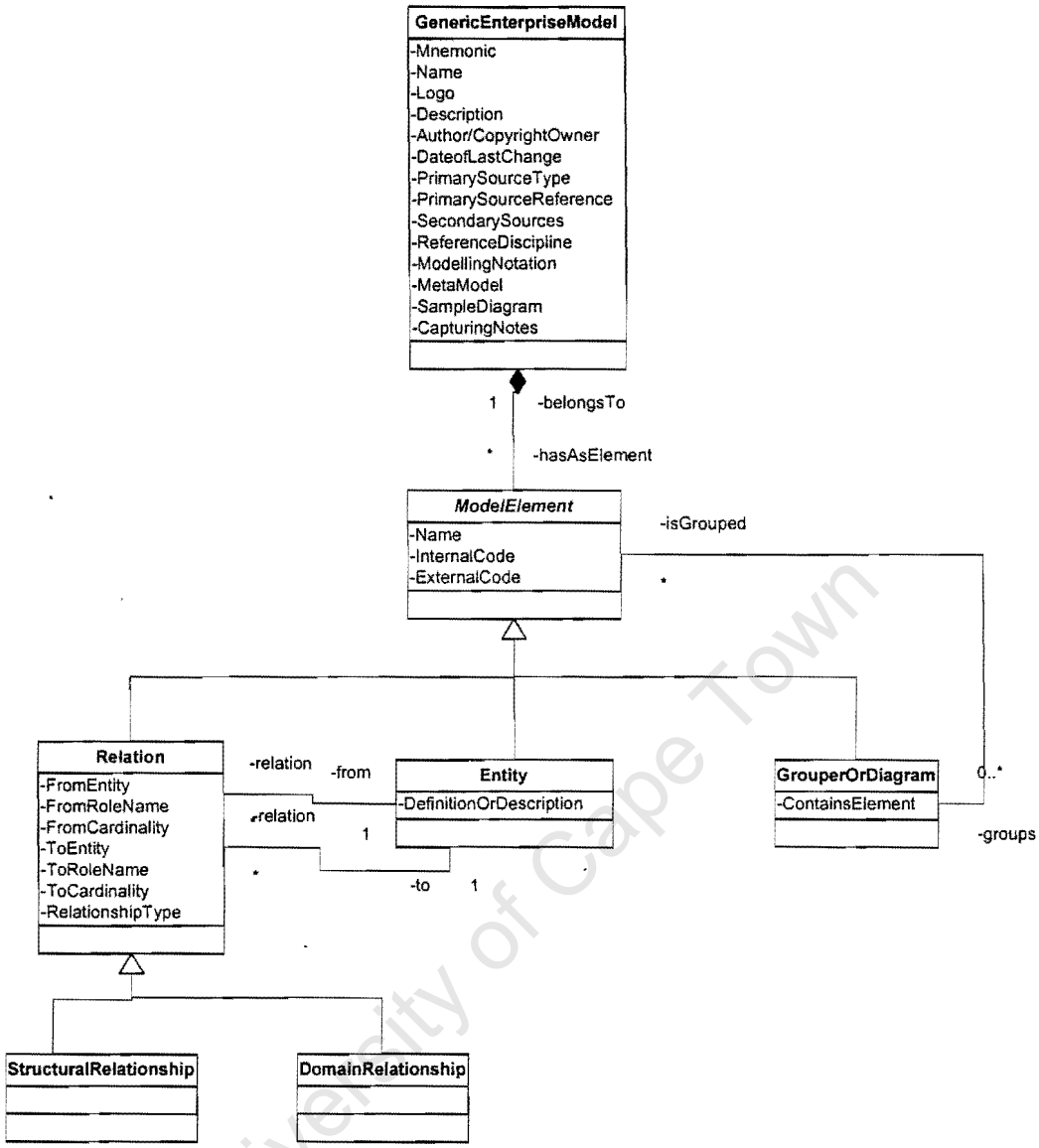


Figure 4-2: Meta-model Used for Model Database

Each enterprise model is characterized by a number of identifying global attributes, such as the model name, its source and author(s) as well as the underlying (semantic) meta-model and diagramming technique.

A model consists of a number of modelling elements (which is an *abstract* type in this meta-model), of which there are three major types. Each modelling element is identified by a name (in some models omitted for some relationships and/or groupers) and an internal code to cross-reference the model elements within the database. Some models also assign explicit (external) codes to each model element. The following are the possible types of modelling elements:

- Entities: usually called entities or objects, they are classes or groups of entities, e.g. "Customer". Many models provide a cursory or detailed description for all or some of the entities in the model. A model has to have a sufficiently large number (ideally 100 or more) of entities in order to be considered for this study.

- Relationships: between entities. The vast majority of the relationships are binary relationships i.e. linking only two entities. A few cases of multiple relationships have been encountered and these were converted to binary relationships in the manner documented in the relevant chapter. Although this does not reflect the full semantics of the model, it simplified technical analysis and framework development significantly.
  Few models identified the *from* and *to* roles, though most models specified a relationship name and cardinalities. In a number of models, a limited few special types of relationships are used. For most models, there are two major types of relationships: semantic structuring relationships (e.g. aggregation and generalization relationships; they are not present in all models) and "domain" relationships which describe real-world dynamic relationships such as "ordersFrom" or "employedBy". The importance of the distinction between these two types is stressed in e.g. [PAZZ98] and will also become evident in the analysis section.
  Although it is not strictly required for each entity to be part of a relationship, the absence of a relationship for a given entity typically means a gap in the domain analysis or model description. Where there are large numbers of unattached entities, a restricted form of the model is usually used including only "connected" entities.
- GrouperOrDiagram: for purposes of representing the model elements in a more easily digestible format, the model elements are normally presented in "blocks" or "units". Where the model is in diagram form, this unit is usually an individual diagram. In the case of textual models (e.g. descriptive models and ontologies) these may be chapters or other types of logical groupings. The critical element of these groupers is their *syntactic* nature i.e. the grouper entity itself is not necessarily domain-related (as is the case for the semantic structuring relationships above). Certain relationship can span two diagrams and therefore belong to two different groupers, and entities can similarly belong to two or more groupers. Not all models use grouper constructs, whereas others use multi-level groupers.

The model database contains a number of enterprise model instances, as described in Chapter 6. The initial capture was done via a variety of tools (including word-processors, text parsers, JAVA compilers and the like), resulting in a series of MS-Excel spreadsheets. These spreadsheets were then consolidated in an MS-Access database which was in turn translated to an XML-encoded text data file. Because of the proprietary nature of some of the models, the full database cannot be made publicly available, although a sub-set (which includes all syntactic information) will be made available on the Web. However, the full dataset is included in electronic format with this thesis for personal research purposes only.

It is important to note that the meta-model above describes the data capture model and therefore it does not include a large number of derived attributes, in particular not the calculated metrics e.g. number of relationships/entities, model complexity etc.

This chapter established the overall methodology for the thesis. It is now possible to develop the analysis framework (next chapter) and build the model database on which the framework will be tested (Chapter 6).

Chapter 5: A Framework for the Evaluation of Enterprise Models

As suggested in the previous chapter, the first step in this research is to develop the analysis framework based on the evaluation criteria available in the literature.

There is a rich body of literature on evaluation criteria for models and related conceptual structures. There are a number of ways in which to structure the discussion of this literature. A possible approach is to make a distinction based on the reference discipline: methodology engineering, systems engineering, ontology research, modelling, etc. An alternative approach is the way in which the criteria are structured: as flat, unstructured lists, as structured trees or hierarchical structures or in the form of a framework based on some theoretical structuring concept or theory. In what follows, a combination will be adopted, using the second approach for the highest-level structuring and the first for the lower-level structuring.

In what follows, the *depth* of treatment of each author's contributions will be fairly shallow. The main purpose is to give a high-level but representative overview of the types of criteria and frameworks that have been suggested in the literature. It is felt that, apart from the substantial space requirements, there would be little value added by explaining each author's contribution in detail. Therefore, the framework structure, the context and definitions used for all the criteria will not be covered, since this information is readily available from the references given. A critical evaluation of only the last few, most relevant, frameworks listed in 5.2 will be given 5.3.1. The meaning or interpretation of the criteria which are suggested in the literature and have been incorporated in the framework, will be fully defined in the relevant sections in chapters 7 to 9. A more systematic, alphabetical list of all suggested criteria (as well as their mapping to the framework dimensions) is found in appendix M.

The focus of the discussion in 5.1 and 5.2, i.e. giving a high-level appreciation of the various criteria from different authors rather than an in-depth discussion, has also led to the decision to *format* the discussion mostly in bullet format rather than the flowing paragraph format more commonly found in dissertations. The latter approach was attempted in an early draft but was found to make the material very inaccessible and defeat the purpose of giving an overview of the diversity of the various evaluation approaches.

5.1 Flat Lists of Evaluation Criteria

Most authors who suggest criteria for evaluating models, do so without ordering, grouping or structuring their list. The following is an overview of some unstructured lists of criteria. They are organised by originating discipline, i.e. evaluation criteria for models, ontologies, methodologies and system engineering. Within discipline, they are generally organised in a chronological order except where similarities dictate a more natural grouping.

5.1.1 Criteria for Evaluating Models

[BENY90:66] suggests the following pragmatic criteria for good (data) models, based in part on Martin & McClure's list [MART85]. According to Benyon, a good model:

- Aids clear thinking
- Is readable by end-users
- Allows for computer/graphical manipulation
- Provides a good basis for communication
- Should be hierarchical i.e. capable of subdivision

- Uses a consistent notation.

The ISO provides a number of criteria for conceptual data models [CROC91], including the following:

- Should be able to model both static and dynamic enterprise aspects
- Should provide a language to communicate the schema to users, and the information processor
- Ease of use and clarity
- Adaptive to changes in the abstraction system
- Implementation independence

[HSU94] mentions the distinction between a model's external validity - how well it correlates with the external reality or domain it is modelling - and its internal validity or consistency: how well it adheres to modelling rules, whether it contains contradictions etc. This distinction is also mentioned by numerous other authors e.g. [CHEN98; BERG00; CROC91].

[WITT94] lists a number of architectural axioms and principles to guide good modelling. They mention the following four axioms:

- Separation of concerns: decomposing the problem into smaller parts.
- Comprehension: remembering that the mind cannot easily handle more than 7 things at a time.
- Translation: the correctness of a design should be independent of its context.
- Transformation: design correctness should not be affected by substitution with equivalent parts.

In addition, they mention the following principles, which are in part derived from the axioms:

- Modular designs by abstraction and high cohesion/loose coupling between modules.
- Portable designs by employing abstract context interfaces.
- Malleable design by modelling the end-user view of the external environment.
- Intellectual control by recording designs as hierarchies of increasingly detailed abstractions.
- Conceptual integrity by uniform application of a limited number of design forms.

Of the latter, Witt suggests that "conceptual integrity is the most important consideration in system design. It is better to have a system omit certain anomalous features and improvements, than to have one that contains many good but independent and uncoordinated ideas." Further in the text (p. 251), two additional criteria to judge architectural quality are added: optimality and robustness.

Some general theoretical criteria for a "good scientific model", gleaned from science philosophers are [VANB96]:

- Occam's razor: use a minimal set of constructs to explain a given situation (construct efficiency).
- Universality.
- Expressiveness.
- Logical completeness.
- Consistent notation.

Chen-Burger formalized the rules and guidelines for model constructions, which allowed for a number of automated model critiques. The following types of critiques are provided [CHEN98]:

- Correctness: detects structural, syntactic and semantic errors
- Completeness: identifies incomplete information in the model and suggests which missing concepts might need to be included.
- Consistency: points out discrepancies in the model.
- Appropriateness: shows deviations from standard practices.
- Alternatives: searches for similar standard models and presents them as alternatives for a given modelling decision. (This is based on a case library of known models.)

She also quotes the recommendation from IBM's BSDM methodology that the depth of an entity model should be no more than four levels deep, to prevent a model from being over-constrained by several layers of dependencies.

Data has a number of quality attributes. These are found in many standard IS textbooks e.g. [VANB99a]. Since a model consists of data, these quality attributes are also applicable to models, though some more than others. The following data quality attributes can also be applied to models [ORLI96]:

- Accuracy (freedom from error).
- Completeness.
- Consistency.
- Validity.
- Quality (as measured by usability and lack of redundancy).

[LOCH98] suggests some proposed standards and "best practices" when developing a corporate data model. Many of these are implementation or notation-specific e.g. relating to the primary key or the use of a specific case for naming entities, tables or relationships. Two more generic criteria are the need for a documentation template and consistency.

Courtot suggests the following questions and analysis elements when looking for packaged data models, although she cautioned that the list is "by no means exhaustive, nor are they all relevant to all situations" [COUR00]:

- Politics.
- Legibility.
- Availability.
- Effect on the business.
- Model quality.
- Architecture/integration.
- Model modifications.
- Depth and quality of coverage.
- Consistency across the model.
- Migration or upgrade complexities.

Claxton and McDougall offer the following 5 yardsticks to evaluate the quality of a data model, indicating that the overall quality of a model is a function of the collective integrity of each of its components [CLAX00]:

- Accuracy.
- Clarity.
- Completeness.
- Conciseness: is the same information repeated?
- Consistency: measures internal contradictions.

Halpin states a number of criteria which he used as the basis for the development of ORM, a language to support Object-Role Modelling. Some of these can be paraphrased to serve as criteria for any type of model [HALP01]:

- Expressibility (including orthogonality).
- Clarity (including non-ambiguity and parsimony).
- Learnability (convenience).
- Semantic stability i.e. minimizes the impact of change.
- Relevance (scope views to just the currently relevant task).
- Validation.
- Abstraction.
- Formality.

In an earlier paper, co-authored with Campbell and Proper [CAMP96], he also emphasized the necessity to decompose models by structuring, clustering and layering.

5.1.2 Criteria from Ontology Research

The set of criteria suggested by Fox, Chionglo and Fadel [FOX93a] in their search for an appropriate representation method for the common-sense model of the enterprise are the following, with the lesser-known concepts explained:

- Generality.
- Competence: how well does it support problem solving?
- Efficiency.
- Perspicuity: is the representation easily understood by the users i.e. self-documenting?
- Transformability: can the representation easily be transformed into another?
- Extensibility.
- Granularity: does it accommodate different levels of abstraction and detail?
- Scalability: does it support large applications?

In a later paper, in dealing with the evaluation of enterprise models, Fox mentioned a few of the above and added the following three [FOX98]:

- Completeness.
- Minimality.
- Precision: is there overlap between concepts or are they partitionable?

Since Fox et al were responsible for developing the TOVE model, it is interesting to contrast their criteria with those used by the "competing" ontology research group, the Enterprise Project, which was responsible for creating the AIAI model. In their quest for the best ontology modelling language, they assigned to each of the candidates a subjective rating on a scale from one to ten for each of the following criteria [MORA94]

- Ease of use, including:
    - Perspicuity to the ontology builder
    - Epistemological proximity
- Expressive power
- Standards
- Translatability
- Transportability
- Formal semantics
- Availability
- Existing user base
- Efficiency / cost
- Flexibility (different ways of expressing something)
- Methodological support.

[VALE96] proposes a set of principles or methodology for constructing a core ontology, and borrows the following four principles from Valente and Breuker:

- Parsimony: use only those concepts which are strictly necessary.
- Clear theoretical basis: i.e. not just a hierarchy of terms, but also a framework (or theory?) of what the domain is about.
- Abstraction: the ontology should not necessarily specify the most common terms but also the basic (abstract or conceptual) categories of the domain.
- These categories should be consistent, complete and make sense in the context of the domain.

The following desiderata for ontologies are equally essential for generic enterprise model development [NOY01]:

- Articulate anticipated usage as well as expected user profiles.

- Use a controlled vocabulary familiar to users.
- Specify mappings between multiple standard vocabularies if they exist.
- Include synonyms.
- Permit extensions.
- Specify semantics (i.e. provide definitions).
- Provide model (search or index) entry using semantics as well as syntactic basis.
- Specify domains and ranges for roles and/or attributes.
- Specify inverse relationships.
- Use some sort of markup language, flags or other mechanism to allow for user views/filters.
- Beware of single-child classes.
- Include verification tools to check for e.g. cycles in class graphs or entities belonging to disjoint classes.
- Allow for lower and upper extensions.
- Include version control.
- Document construction considerations/standards and give guidelines for extensions.

[SWAR96] specifies some other desired ontology characteristics, useful from a lifecycle (management) viewpoint:

- No representational commitment (e.g. compulsory attributes, constraints etc) to retain maximum flexibility.
- Extensibility.
- Scope driven by identified user needs.
- Can be integrated with other models.
- Should have an overall organizing principle.

[GOME99] summarized ontology design criteria and principles that have been proved useful in the development of ontologies, of which the following apply to models:

- Clarity and Objectivity, which means that the ontology should provide the meaning of defined terms by providing objective definitions and also natural language documentation.
- Completeness, which means that a definition expressed in terms of necessary and sufficient conditions is preferred over a partial definition (defined only through necessary or sufficient condition).
- Coherence, to permit inferences that are consistent with the definitions.
- Diversification of hierarchies to increase the power provided by multiple inheritance mechanisms.
- Modularity to minimize coupling between modules.
- Minimization of the semantic distance between sibling concepts which means that similar concepts are grouped and represented using the same primitives.
- Standardization of names whenever is possible.

5.1.3 Criteria for Evaluating Methodologies and Related Frameworks.

Avison & Fitzgerald [AVIS95:435-437] have an extensive list with areas of concern when comparing methodologies, and suggest a large set of requirements for the design of a methodology. Many of these are based on other authors although some of the original unpublished sources are difficult to obtain. The following extracts only those requirements which could also be considered applicable to models.

From Catchpole [CATC87], who summarized the views of a number of other authors in his PhD dissertation:

- Documentation easily understandable by user & analysts.
- Separate logical and physical designs.
- Design validity checks for inconsistencies, inaccuracies and incompleteness.
- Easy change.

- Teachability of method.
- Draw/model the boundary between what can/cannot be computerized explicitly.
- Design for future change.
- Effective communication.
- Simplicity.
- Ongoing relevance.
- Automated development aids.
- Consideration of user goals and objectives.
- Systematic way of looking into the future to incorporate possible future changes.
- Integration of technical and non-technical systems.

From Bjørn-Andersen [BJØR84]:

- What research paradigms/perspectives form the foundation?
- What is the underlying value system?
- In which context is the methodology useful?
- Is modification enhanced or possible?
- Does communication & documentation operate in the user's dialect (expert or not)?
- Transferability?
- Does it deal with the social environment (e.g. conflict, change management)?
- Is user participation *really* encouraged?

From Bubenko [BUBE86]:

- Theoretical investigations of concepts, languages etc.
- Practical experiences/cases of method application.
- Cognitive-psychological investigations.

Jayaratna similarly lists an exhaustive literature survey of criteria to evaluate methodologies [JAYA94]. In addition to some of the ones mentioned above (including Martin & McClure), he cites the following:

- Purpose.
- Manner & ease of production.
- Ease of use.
- Ease of maintenance.

[HUTT94] describes the best-known Object-Oriented Analysis & Design (OA&D) methods available at that time. Although he does not provide a comparative analysis, his description aims to be comparative by providing the following common elements of each method:

- Method name.
- Name & affiliation of author.
- Introduction: overview of method.
- Purpose and scope: defines the role of the method in the total development cycle and its scope.
- Concepts: defines the concepts used within the method. The concepts are described in terms of commentary on the technical framework plus descriptions of the concepts specific to the method.
- Deliverables: gives examples of the types of the output when using the method.
- Process: describes the types of activities that are typically performed in order to produce the deliverables.
- Techniques: describe the techniques that are used to develop the deliverables.
- Pragmatics: describes how the method is delivered to developers and any tools that support it.
- Further information: section must contain a list of sources that have been used to compile the description of the method. Ideally this source material should be publicly available.

In a separate publication, [FRAN99a] places the following requirements on modelling languages. These also apply to models:

- Formality.
- Completeness.
- Simplicity.
- Correctness.

Vernadat compares a large number of *enterprise modelling languages* for the purposes of the ESPRIT project using the following criteria [VERN97]:

- Formality: does the language use *formal* constructs?
- Modelling principles: are the following modelling principles supported?
    - Separation of concerns
    - Functional decomposition
    - Model genericity
    - Separation of functionality and behaviour
    - De-coupling of process and results
- Modelling views: are all of the following views/aspects modelled: function/control, information, resource, organization and human?
- Modelling support: is there a supporting methodology and CASE tool?

[VANH99] uses four characteristics to compare meta-modelling studies:

- Scope (in the sense of size).
- Basis or sources.
- Degree of formality.
- Visibility (how can outsiders deliver input) and validity of the building process (to assure correctness, completeness and usability).

These are in fact very much a subset of the criteria used by [BRIN96] for comparing four method engineering languages:

- Scope: the range of applications for which the language is being used.
- Basis: underlying modelling language or formalism.
- Paradigm: the underlying philosophy or "way of thinking".
- Explicitness: the explicitness of the method descriptions.
- Focus: whether the method's products or process are emphasized.
- Size: the average size of a method specification using that language.

Williams [WILL96] gives a detailed list of 45 requirements for an Enterprise Reference Architecture (ERA). The following is a selection of those that are also applicable to models. Models should:

- Use a minimal set of basic building blocks, but these should be able to show all existing relationships.
- Be used to develop or compare against standards.
- Promote the concepts of flexibility, modularity and adaptability.
- Explicitly show the place of the human in the enterprise, functionally and organizationally.
- Enable reuse of previous designs.
- Apply formal methods by means of rigorously defined syntax and semantics.
- Provide a common basis for discussion and interpretation.
- Reflect the decision making process.
- Be amenable to flexible modification.
- Have supporting guidelines.
- Be independent of existing technology, system configuration or implementation.
- Accommodate multiple views.
- Limit complexity to facilitate human comprehension and computational load.
- Allow for modularity, substitution and high cohesion/low coupling.
- Be open-ended and extendable.

5.1.4 Criteria from OO and SE

An excellent paper by Korson and McGregor [KORS92] suggests 23 criteria for OO class libraries. Each of the criteria is ranked according to the following set of desirable attributes: completeness, consistency, ease of learning, ease of use, efficiency, extendibility, integratability, intuitiveness, robustness and support. Apart from a number of technical criteria (partial functions, exception handling, interface specification etc.), most of their criteria are perfectly applicable to models since there is but a small step from a class library to a class diagram. A model should:

- Be complete (cover the complete domain).
- Be designed around a few key abstractions.
- Model standard knowledge.
- Use generalization/specialization relationships (inheritance structure).where necessary
- Be pure i.e. all classes should be related; no stand-alone data.
- Be loosely coupled.
- Be efficient, given the time and space constraints
- Be consistent in naming and defining concepts.
- Provide generic classes where possible.
- Be full implemented or indicate the degree of completeness.
- Have the documentation organized in the same way as the model.
- Have documentation that provides a high-level overview of both model structure and content.
- Have documentation that is available for the different user levels.
- Have indexed documentation with alphabetical, hierarchical and keyword entry points.
- Have a formal specification for the model components.
- Come with tools for accessing the models.
- Come with tools for extending the model.
- Have support.
- Be updated regularly.

[LYON98] suggest three criteria for OO models - with some associated metrics as given by Lorenz & Kidd:

- Low class coupling.
- Inheritance tree not too deep (Lorenz & Kidd suggest no more than 6 levels).
- Not too many operations per class (Lorenz & Kidd suggest no more than 20 operations).

A typical list of software quality characteristics is: completeness, complexity, correctness, generality, integrity, modularity, portability, reliability, redundancy, and time efficiency or storage efficiency as well as workmanship. [HAUS92]

Hewlett-Packard developed a set of software quality factors which have been popularized under the acronym FURPS [PRES97c]:

- Functionality including feature set, generality and security.
- Usability comprising human factors, overall aesthetics, consistency and documentation.
- Reliability measuring various technical failure and error rates as well as accuracy.
- Performance including processing speed, resource requirements and efficiency.
- Supportability consisting of maintainability (which can be specified into extensibility, adaptability, serviceability), configurability, testability and compatibility.

Many of these have been refined in the ISO's attempt to standardize the terminology used in the context of software quality as per its standard 9126 released in 1991 (updated in 1998) on "Information Technology-Software Product Quality". This specification focuses on six broadly defined and described quality characteristic relating mainly to finished software products [BEVA95].

The following list, taken from a draft software quality process analysis proposal from the NIST/IEEE, summarizes the software qualities defined in the ISO standard [ISO98]:

- Functionality: suitability, accuracy, interoperability, security and compliance.
- Reliability: maturity, fault tolerance, recoverability and compliance.
- Usability: understandability, learnability, operability, attractiveness and compliance.
- Efficiency: time behaviour, resource utilization and compliance.
- Maintainability: analyzability, changeability, stability, testability and compliance.
- Portability: adaptability, installability, co-existence, replacability and compliance.

Wallace and Reeker also give a nice overview mapping all of the above to various other models on software quality. [WALL01] Because not all of the software quality factors are applicable to models, and all of the relevant quality factors are subsumed in the two lists above, the earlier works are not listed here to avoid repetition.

5.2 An Overview of Evaluation Frameworks

The following represents a selection of frameworks for evaluating the quality of modelling approaches and methodologies. The difference with the earlier set of criteria is the provision of a structure i.e. an overall framework which organises the different criteria. Many of these frameworks can be partially adapted to evaluate also the outcome or product of a methodology or modelling process, namely the resultant enterprise model. Many more frameworks have been proposed but there is a high degree of overlap between many of them. The following frameworks are those from which the proposed model quality evaluation criteria were drawn. The first three are matrix presentations of software quality factors, presented in a chronological order. The last few frameworks are modelling methodology evaluation frameworks that are based on some more theoretical organising principle and are also presented chronologically to highlight the influences and developmental progression of each framework.

5.2.1 McCall's Quality Factors / the GE Model

McCall et al proposed a comprehensive categorization of factors that affect software quality. They distinguish clearly between the quality factor and the quality metric which can be measured directly or indirectly. Table 5-1 lists both, as well as the correlation between them [PRES97c; GILL97a].

Note how the factors are grouped into 3 major categories: (software product) revision, transition and operation.

Table 5-1: McCall's Quality Factors [PRES97c; GILL97a].

	OPERATION					REVISION			TRANSITION		
Software quality metric / Quality factor	Correctness	Reliability	Efficiency	Integrity	Usability	Maintainability	Flexibility	Testability	Portability	Reusability	Interoperability
Auditability				x				x			
Accuracy		x									
Communication commonality											x
Completeness	x										
Complexity		x					x	x			
Conciseness			x			x	x				
Consistency	x	x				x	x				
Data commonality											
Error tolerance		x									x
Execution efficiency			x								
Expandability							x				
Generality							x		x	x	x
Hardware independence									x	x	
Instrumentation				x		x		x			
Modularity		x				x	x	x	x	x	x
Operability			x		x						
Security				x							
Self documentation						x	x	x	x	x	
Simplicity		x				x	x	x			
System independence									x	x	
Traceability	x										
Training					x						

5.2.2 The Böhm Model

A fairly similar model was proposed by Böhm, which contains very much the same characteristics, although there are a number of semantic nuances. The model has three levels, with the intermediate level further split into primitive characteristics which can be measured [BÖHM76].

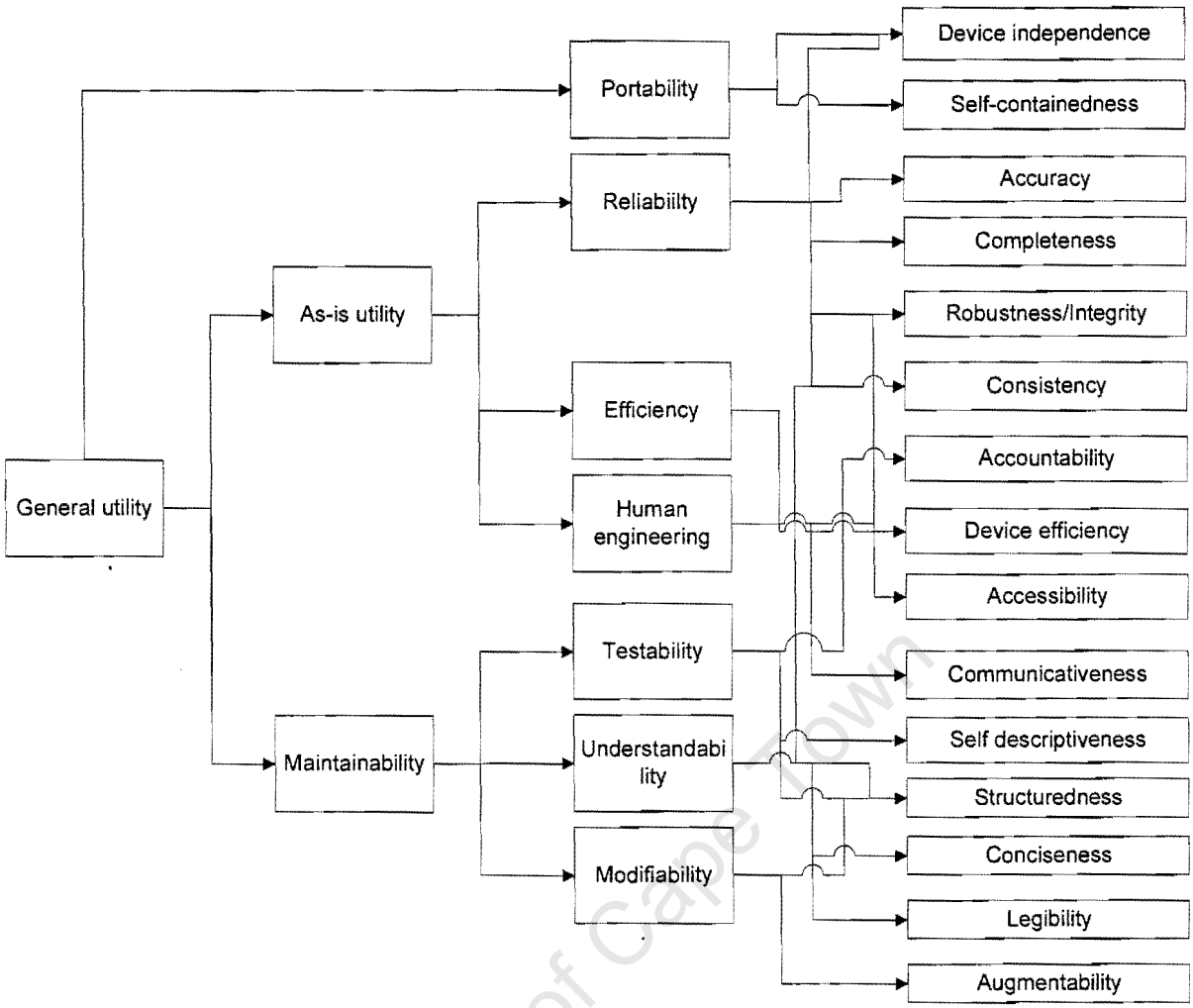


Figure 5-1: Böhm's Quality Model [BÖHM76].

Note that the emphasis of Böhm's model is, similarly to McCall's, on the more technical criteria. Hyatt summarizes both of the above and maps their criteria against those mentioned in the ISO 9126 standard, which defines and describes the software product quality criteria more formally [HYAT96].

Table 5-2: Comparison of Software Quality Models [HYAT96].

Criteria/Goals	McCall, 1977	Boehm, 1978	ISO 9126, 1993
Correctness	X	X	maintainability
Reliability	X	X	X
Integrity	X	X	
Usability	X	X	X
Efficiency	X	X	X
Maintainability	X	X	X
Testability	X		maintainability
Interoperability	X		
Flexibility	X	X	
Reusability	X	X	
Portability	X	X	X
Clarity		X	
Modifiability		X	maintainability
Documentation		X	
Resilience		X	
Understandability		X	
Validity		X	maintainability
Functionality			X
Generality		X	
Economy		X	

He uses these to build his own quality framework. Very interesting are his suggested metrics for measuring some of these quality attributes. Two of these are also used in this research: document structure and readability index.

5.2.3 Gillies' Hierarchical Model of Quality

Gillies surveyed six large organizations to elicit hierarchical models of quality by asking software users and developers for their quality criteria. He organized this in the schematic model as per Table 5-3, which highlights the conflict between users who are seeking software that is fit for a purpose as opposed to developers who aim at conforming to the development specifications [GILL97a].

Table 5-3: Gillies' Hierarchical Model of Quality [GILL97a].

Quality = correctness	
Technical Factors "conforms to specification"	**Business factors "fit for purpose"**
Reliability	Added value
Maintainability	Cost
Integrity	Timeliness of delivery
Efficiency	User satisfaction
Usability	Ease of transition
Adaptability	
Interoperability	
Portability	

This is based on a larger set of criteria. Gillies points out that these criteria are correlated as per Table 5-4.

Table 5-4: Correlation between Gillies' Quality Factors [GILL97a].

Gillies 0 = no correlation + = positive corr. - = negative corr.	Reliability	Efficiency	Integrity	Security	Understandability	Flexibility	Ease of interfacing	Portability	User consultation	Accuracy	Timeliness	Time to use	Appeal	User flexibility	Cost/benefit	User friendliness
Reliability		0	+	+	0	0	0	0	0	+	-	+	+	0	+	+
Efficiency	0		-	-	-	-	0	-	0	0	0	+	+	-	+	0
Integrity	+	-		+	0	0	+	0	+	+	-	0	0	0	+	0
Security	+	-	+		0	0	+	0	0	+	-	0	+	0	+	0
Understandability	0	-	0	0		+	+	0	+	+	0	-	+	0	+	0
Flexibility	0	-	0	0	+		+	+	0	0	-	-	+	+	+	0
Ease of interfacing	0	0	+	+	+	+		+	0	0	-	-	+	0	0	0
Portability	0	-	0	0	0	+	+		0	0	-	0	+	+	0	0
User consultation	0	0	+	0	+	0	0	0		+	-	0	+	+	+	+
Accuracy	+	0	+	+	+	0	0	0	+		-	0	+	+	+	0
Timeliness	-	0	-	-	0	-	-	-	-	-		+	+	0	+	-
Time to use	+	+	0	0	-	-	-	0	0	0	+		+	-	+	0
Appeal	+	+	0	+	+	+	+	+	+	+	+	+		0	0	+
User flexibility	0	-	0	0	0	+	0	+	+	+	0	-	0		+	+
Cost/benefit	+	+	+	+	+	+	0	0	+	+	+	+	0	+		0
User friendliness	+	0	0	0	0	0	0	0	+	0	-	0	+	+	0	

This is an expansion of the correlation matrix suggested by Perry as per diagram below [GILL97a].

Table 5-5: Correlation between Perry's Quality Factors [GILL97a].

Perry 0 = no correlation + = positive corr. - = negative corr.	Correctness	Reliability	Efficiency	Integrity	Usability	Maintainability	Testability	Flexibility	Portability	Reusability	Interoperability
Correctness		+	0	0	+	+	+	+			
Reliability	+		0	0	+	+	+	+		-	
Efficiency	0	0		-	-	-	-	-	-	-	-
Integrity	0	0	-		+					-	-
Usability	+	+	-	+		+	+	+			
Maintainability	+	+	-		+		+	+	+	+	
Testability	+	+	-		+	+		+	+	+	
Flexibility	+	+	-		+	+	+			+	
Portability			-			+	+			+	+
Reusability		-	-	-		+	+	+	+		
Interoperability			-	-					+		

5.2.4 Avison & Fitzgerald's Framework for Methodology Comparison

Avison & Fitzgerald [AVIS95:446-448] have developed the following framework for comparing methodologies, based on a number of previous attempts and other authors such as Wood-Harper. Although their approach resembles a hierarchical structure, the authors describe it as a framework because it takes contextual and philosophical considerations into account. These considerations

include academic methodology taxonomies, and the actual evaluation criteria for each element depend on the methodology under consideration. Each element is elaborated on in much detail in the text.

Their seven basic framework elements, with sub-elements, are:

1. Philosophy:
    a. Paradigm: (hard) science versus (soft) systems (see also [HIRS89]).
    b. Objectives.
    c. Domain.
    d. Target: particular types/sizes of organizations?
2. Model (verbal / analytical / iconic / simulation).
3. Techniques and tools.
4. Scope (life cycle, level of detail).
5. Outputs.
6. Practice:
    a. Background: academic or practioner/commercial.
    b. User base: numbers & types of users.
    c. Players: users and/or analyst.
7. Product: Does it include: software? Documentation? Training? Telephonic/online help?

They also suggest two possible additional elements:

- Quantity of specifications & documentation.
- User modifiability.

5.2.5 The Seligman Framework

The Seligman framework for analysing information systems development methodologies proposes that there are five ways or elements that should be considered when analysing Information Systems Development Methodologies (ISDM) in general [HOMM98].

Figure 5-2 illustrates the structure of the Seligman Framework. Included within the original framework is the way of designing which encompasses both the way of modelling and the way of working.

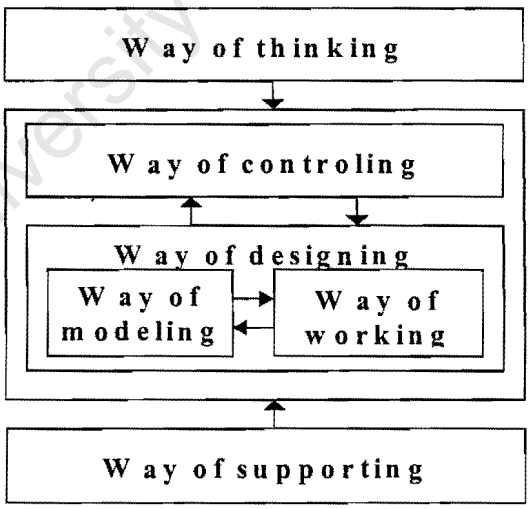


Figure 5-2: Seligman's Framework for understanding IS Development Methodologies.

Each element can be described as follows:

- The way of thinking refers to the (often-implicit) assumptions applied to the perception of the empirical object (ontology), and to the way this can be studied or designed (epistemology).
- The way of modelling describes the models used and their relationship, as used during systems development.

- The way of working describes the tasks and sub-tasks that must be carried out during the development of the models. This views modelling as a process-based activity which occurs over a period of time.
- The way of controlling is concerned with the monitoring of the activities (project management tasks) described in the way of working.
- The way of supporting refers to the tools that support the systems development.

5.2.6 Brazier's Framework for comparing Modelling Methodologies.

Brazier considers three broad approaches for comparing modelling frameworks [BRAZ98]:

- Problem-oriented i.e. based on the approach to a given problem.
- Purpose-driven i.e. based on the aims behind the framework.
- Based on the modelling primitives allowed.

In an attempt to integrate these three broad approaches, Brazier then suggests the following framework (with associated criteria) to compare ontologies:

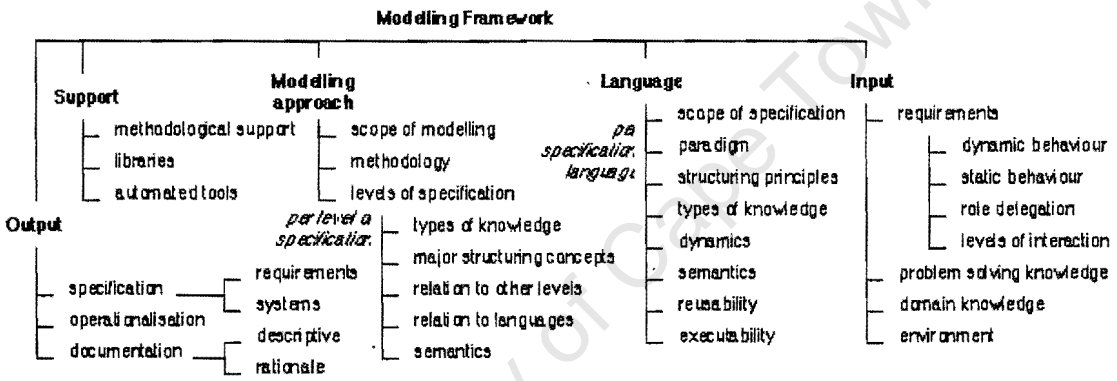


Figure 5-3: Brazier's Framework for Comparing Modelling Frameworks.

One of the interesting aspects of this framework is that these criteria have been developed in the context of ontology research i.e. outside the normal information systems development paradigm.

5.2.7 Hommes' framework

Bart-Jan Hommes' [HOMM98] framework for analysing the quality of a business modelling technique was used as a basis for developing Price & Van Belle's framework below. Hommes bases his framework in turn on Seligman's framework. In his framework, the modelling technique is influenced by two elements: the way of modelling and the way of working.

The way of modelling is influenced by the level of expressiveness, which can be described as how well the conceptual system corresponds to the business system. The way of working is influenced by the level of arbitrariness, which is the degree of freedom the modeller has when modelling a given domain.

The two elements are in turn influenced by four different properties of the resulting model: consistency, correctness, comprehensibility, and usability. The expressiveness of a modelling technique is affected by the consistency and correctness of the model. These are described as follows:

- Consistency is the overlap in modelled concepts between the individual sub-models of a modelling technique. A certain degree of overlap is necessary in order to obtain a coherent model

cycle while too much overlap increases the danger of the models becoming inconsistent. Thus a fine balance needs to be struck to ensure that the overlap is sufficient but not overwhelming.

- Correctness refers to the way in which the model describes the corresponding business system i.e. the model should be a clear representation of the business system.

The arbitrariness of a modelling technique is affected by the correctness, comprehensibility and usability of the model. These are described as follows:

- Correctness relates to the determinism (the degree of freedom) of the modelling technique. When the models do not describe the corresponding business system in a clear way, the result is that several different models can describe the same business system, increasing the arbitrariness.
- Comprehensibility also leads to determinism. Modellers that do not fully comprehend the business system and its representation can be forced into choices between options on arbitrary motives and thus towards arbitrariness.
- Usability influences the arbitrariness of a modelling technique since complex models (models that demand a lot of knowledge from a modeller) might not be fully comprehended by the people using them. This will also lead to forced choices between options on arbitrary motives.

The framework is depicted in Figure 5-4

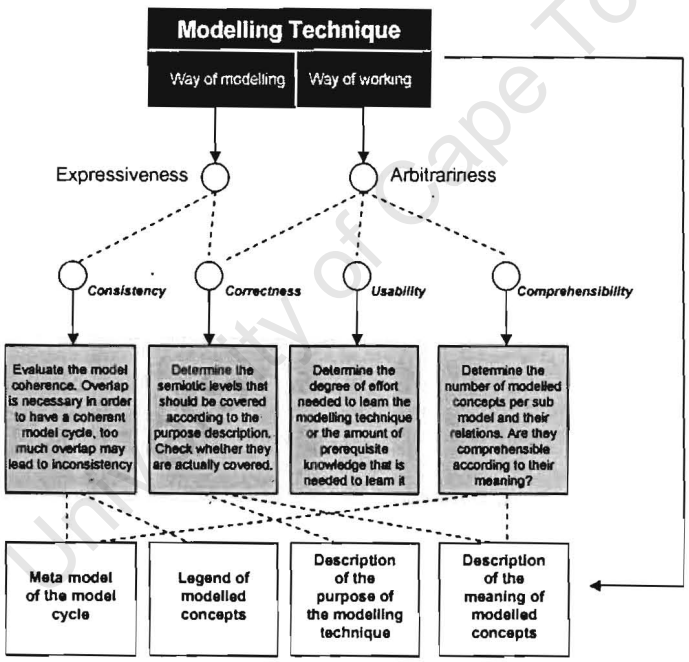


Figure 5-4: Hommes' Framework for Analysing the Quality of a Business Modelling Technique.

5.2.8 Price & Van Belle's Framework

As a result of some earlier research, we published the following "first-generation" framework, based on Seligman's and Hommes' model but supplemented with some criteria from Fox and Williams.

It is believed that Hommes' model for evaluating techniques can also be used to evaluate the final products of methodologies i.e. to evaluate the quality of their outputs: (generic) enterprise models. Three requirements were added to the framework to extend its evaluation criteria from those of a quality modelling technique to a generic model [VANB00].

Efficiency.

There is more than one way to represent or model the same knowledge, each representation is of a differing complexity. This also relates to the degree to which the modelling technique is capable of improving the productivity of design and of ensuring that there is a degree of consistency applied to all the models. An efficient modelling technique should use the available resources to leverage generality.

Since an efficient model would be correspond more to the business system, the efficiency attribute of the enterprise reference architecture would be evaluated as being an aspect of the Way of Modelling, and would thus fall under the expressiveness property.

Generality (or genericity).

GERAM notes that many of the existing ERP solutions and architecture methodologies are industry specific (such as SAP and CIM-OSA for the manufacturing industry), and specifies that an enterprise reference architecture should, by definition, be applicable to all organizations and encompass all industries and sectors.

If the aim of a generic enterprise reference architecture is to model business concepts that are applicable to all business systems, then generality would be defined as being a Way of Modelling aspect of the modelling technique, and consequently a characteristic of the technique's arbitrariness.

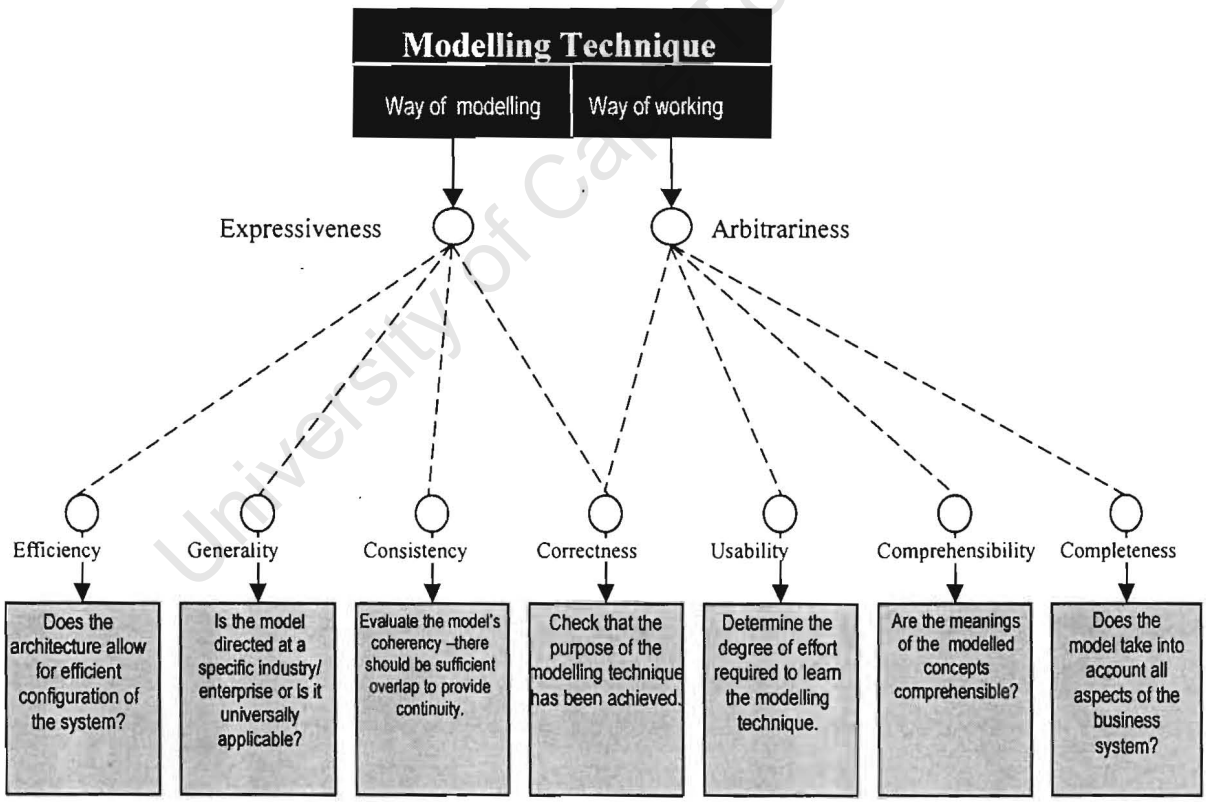


Figure 5-5: Price and Van Belle's Early Framework [VANB00].

Completeness.

The generic enterprise reference architecture should encompass all aspects of the organization. The methodologies that were investigated emphasized this aspect to varying degrees, and used different approaches to being a 'complete' architecture.

PERA places more emphasis on the human involvement in the business system than other architectures. GERAM requires that a generic architecture should consider a business system from the

following three aspects: the human factors, the customer product or service aspect, and the information systems aspect. In considering the human aspect, GERAM elaborates on the importance of including the human perspective: the tasks, skills, structures and other human-related factors that form part of the holistic business system.

The completeness of the architecture is indicated by the various views that the modelling methodology incorporates: CIM-OSA uses an organizational view, a resource view, an information view, and a functional view; the R/3 architecture uses a process view, a data view, an information flow view, a function view, and an organizational view.

The completeness attribute would be defined as a prerequisite of the modelling technique, and would consequently be grouped under the arbitrariness property of the Way of Working.

5.3 The Proposed Framework for Evaluating the Quality of Enterprise Models

The proposed framework will be developed in the following sequence. Firstly, the shortcomings of the frameworks mentioned above will be discussed briefly. Next, the two major dimensions – equivalent to the conceptual structuring principles – for the proposal will be explained and motivated. Then, the framework will be "populated" with criteria. Finally, the selected criteria will be cross-referenced against the full list of criteria, collected from the survey presented earlier in this chapter.

5.3.1 Problems with the Current Frameworks

There are a number of problems with the application of the frameworks mentioned earlier. To be fair, it must be clearly stated that these problems are not intended as criticism since the frameworks are taken out of the context for which they were developed. The problems therefore apply only when attempting to adopt these frameworks in the new, different context of modelling.

Three problems appear to apply to virtually all of the frameworks mentioned in section 5.2.

- ***The Grounding Problem.*** All lack an underlying theoretical basis: although the distinctions may be based on a natural grouping or structuring of the desired factors but there is no underlying theoretical or philosophical basis for the framework dimensions. Although it is acknowledged that the frameworks may still be valuable and valid – as long as they are based on the principles of soundness and completeness – authors such as [FRAN98b] stress the value and importance of a strong theoretical grounding for an evaluation framework. The proposed framework will root itself in the discipline of semiotics and draw on a fundamental distinction which has proved valuable in many different contexts.
- ***The Partial Applicability Problem.*** Most of them apply only partially to the evaluation of enterprise models. A lot of criteria relate to the development process, which is not applicable in our case. This is not an intrinsic criticism but refers to the lack of a suitable framework which has been developed in the context of modelling.
- ***The Lack of Generality Problem.*** None of the frameworks is generic in such a way that they could be applied to evaluate the quality of similar "intellectual works" in IS or any other discipline i.e. a given framework (say for measuring the quality of software) and can generally not be used for evaluating, for instance, models, systems architectures, products, or artworks. This is partly due to their lack of theoretical grounding. Although some frameworks have a limited applicability or transferability to related domains, it is the intention that the framework suggested in this research will be more generic in scope.

There are a number of more specific problems with each of the frameworks examined in section 5.2. Perhaps of most interest is the rejection of the framework developed earlier i.e. the Price-Van Belle framework. The major criticisms of the framework are the following (with thanks to fellow researchers for their contributions):

- The distinction between "way of modelling" and "way of working" makes far less sense when applied to the output of modelling activities than when used for modelling methodologies. The models themselves often have only faint signatures left from the way of working and modelling.
- The jump from "way of modelling" to "expressiveness", and the parallel link from "way of working" to "arbitrariness" are not intuitive and are subject to challenge.
- The links between the second (expressiveness and arbitrariness) and the third level criteria are similarly somewhat debatable. Of particular concern is the fact that "correctness" is bound to *both* higher-level classifications, which introduces an immediate validity question.

Thus the challenge is to develop a framework for evaluating enterprise models that has sound theoretical foundations, is specific to the evaluation of enterprise models yet can be ported to similar domain or problem areas.

5.3.2 Framework Dimensions

The first and major dimension is grounded in linguistics, information and communication theory and semiotics. The key distinction used in the framework is the fact that all informational objects have a syntactic, semantic and pragmatic aspect.

- The syntax refers to the type of constructs and the legal ways of combining them i.e. the physical or logical representation of the information. In a model, this refers to the meta-model entities. A syntactic model analysis will therefore be concerned with the number and types of entities, relationships etc.
- The semantics refers to the meaning i.e. the sense of the information by interpreting the token or signifier.
- Pragmatics refers to the context i.e. considerations, issues and background information which influences or moderates the interpretation of the information.

Table 5-6: Main Classification Dimension of Proposed Analysis Framework.

Classification concept	Related terms and mappings
Syntax	Symbols, form, shape, structure
Semantics	Meaning, denotation, sense
Pragmatics	Background, situation, context

This distinction allows the unambiguous classification of any analysis technique.

The *syntactic* model analysis is concerned with the purely structural aspects of the model, regardless of the underlying meaning of the model and its elements. Most of the analysis will be derived from the hard sciences i.e. the disciplines of software engineering, computer science, graph theory and network analysis. This includes a large variety of standard syntactic metrics relating to size, grouping, layering, inheritance structure, and network visualization, as well as some less standard metrics such as interface aesthetics. Syntactic analysis tends to be fairly exact and mathematical in nature since it is based on numerical formulae.

The *semantic* analysis of models is concerned with the intrinsic *meaning* of the model i.e. its relationship and mapping to the underlying domain reality it is representing. The syntactic analysis is content to deal with the structural relations, i.e. shape and form of the entities and their relationships and groupers, and therefore treats all entity and relationship names as mere "alphanumeric labels". The essence of semantic analysis is to unravel the *meaning* of the name (label, word, token) used for a specific model element (entity, relationship, grouper). Put another way, semantic analysis is concerned with the correspondence (mapping, projection, validity) between the model (abstract or intellectual construct) and the underlying domain (reality). Whereas syntactic analysis is fairly technical and easy to automate, semantic analysis involves the more tricky matters of meaning and interpretation and thus lends itself not quite as easily to objective and/or automated analysis. The reference disciplines for this type of analysis is linguistics and lexicography, while much of the analysis is concentrating on similarity, correspondence and cluster analysis.

Pragmatic model analysis, as defined in the framework used for this research, is concerned with those metrics and criteria which cannot be assessed purely on the basis of the information contained within the model, but which requires the consideration of information regarding the use, environment or context of the model i.e. information outside the model. The analyses techniques falling under this heading include the face validity, degree of use, authority of model author, availability, cost, flexibility, adaptability, model currency, maturity and support. Most analysis relies on the searching and ranking of certain specific information details, often involving a degree of subjective interpretation and an understanding of commercial business issues.

This distinction has been around for well over a century and has been well validated in a number of disciplines. Quite a few of the researchers who have been involved with the development of information theory and the philosophical foundations of information systems, have referred to this distinction [see e.g. BENY90; STAM95]. It is therefore very surprising that this has not been applied as such to evaluation frameworks in the field of models, which are "complex information products" *par excellence*. As will be shown in the rest of this research, the classification system is particularly useful since it allows one to classify the metrics for the quality factors as well.

A second dimension is not as clear-cut and is presented here more for future research purposes, since it will not be explored further in this thesis. It is proposed that, for each category above, there is also a distinction between "absolute" and "relative" measures i.e. for some quality factors, "more is better" whereas for others, the ideal value depends on factors not directly related to the intrinsic nature of the model. It must be recognized that this distinction is not always as clear as it may seem: the distinction is actually more a matter of degree and there is a fuzzy cross-over area in the middle where it may be somewhat more difficult (or arbitrary) to decide whether a given quality measure is absolute or relative.

Examples of the distinction for "syntax measures" are the following:

- Model *correctness* (Is the model free of syntax errors? Are the various modelling constructs used in a consistent manner?) is an absolute measure: the more correct a model is, the better.
- At the other extreme, the *architectural style* of a model is clearly a *relative* measure, since it involves subjective evaluation and weighting of the various stylistic features. It remains a syntactic measure since it is purely based on evaluating the form, shape or structure of the model.
- Model *complexity* is also a relative measure: a very simplistic model is not likely to be of high quality, but a very complex model is not necessarily desirable either. Despite this uncertainty,

however, model complexity in itself *is* an important indicator of model quality; it is just that the interpretation of, or yardstick for, the complexity depends on factors not related to the model.

In practice, it will be found that the distinction on which the second dimension is based, is partly parallel to the observations and subsequent research done by Gillies, and partly based on time-honoured philosophical principles. Gillies emphasized the dichotomy between the software developers, who adopt a technical view, and the business users, who adopt a very different applied perspective [GILL97a]. This tension is well-known in the discipline, and is really just indicative of the overall difference in disciplinary perspectives between computer science and information systems. Gillies therefore classified his quality measures under the following two headings:

- Technical factors: "conforms to specification"; and
- Business factors: "fit for purpose".

It is hereby proposed that this distinction is even more fundamental than the traditional tension between technology and business: it is the dichotomy of theory versus application, scientist against practitioner; or, to put it in more philosophical terms: the intrinsic versus extrinsic qualities. Table 5-7 develops the distinction somewhat further.

As mentioned earlier, it has long been realized in philosophy that this distinction is not as clear-cut as it may seem. The boundary between intrinsic and extrinsic is fuzzier than it seems at first inspection. This is discussed more fully in e.g. [WEAT02] or any philosophical treatment of e.g. Kantian classification schemes.

Table 5-7: Possible Second Dimension for Proposed Analysis Framework

Classification concept	Related terms and mappings
Absolute measures	Theoretical; "das Model an Sich"; the model as object; objective standards; intrinsic qualities; technical factors; "Conforms to specification"; computer science; academic.
Relative measures	Applied; "das Model für Uns"; the model as subject; subjective standards; extrinsic qualities; business factors; "Fit for purpose"; information systems; practitioner.

The following notes are important when considering the proposed classification scheme:

- The two dimensions are not fully orthogonal: relative, subjective qualities by definition involve an outside perspective and hence there is scope for a possible overlap with the "pragmatic" cells; or, conversely, absolute measures cannot be pragmatic. Although the dimensions are not necessarily 100% orthogonal, they still serve a useful purpose. As the examples above (for the some syntactic factors) demonstrate, this distinction works quite well in practice: there is a difference between the setting of the standards against which the quality factors will be measured (the second dimension), and the actual domain from where the factors are derived (the first dimension).
- Neither of the dimensions is specific to modelling or even information science, so the "skeleton" of the framework (i.e. the classification scheme) could easily be ported to other disciplines or domains. On the other hand, a second-stage "filling in" of the framework cells makes it apply specifically to generic models.

It must be emphasized strongly that, for purposes of this research, the second dimension is not considered of primary importance, and it should be interpreted as more speculative than normative.

This research will focus on the framework using the first dimension: syntactic versus semantic versus pragmatic measures.

5.3.3 Populating the Framework Skeleton with More Detailed Criteria

Table 5-8 lists the proposed quality factors within the framework structure. Many factors are of a composite nature or clusters and can be sub-divided into sub-factors or topics.

It is important to note that the criteria listed here are simple evaluation criteria. Measuring the overall "quality" of a model is not possible within the proposed framework because model quality represents a composite concept including many of the above criteria, with a weighting very much dependent on the actual purpose of the analysis.

As seen above, the term "quality" encompasses most of the above criteria:

> "Finally, we concluded that calculating and understanding the value of a single overall metric for software quality may be more trouble than it is worth. The major problem is that many of the individual characteristics of quality are in conflict; added efficiency is often purchased at the price of portability, accuracy, understandability, and maintainability; added accuracy often conflicts with portability via dependence on word size; conciseness an conflict with legibility. Users generally find it difficult to quantify their preferences in such conflict situations" [BÖHM78, p. ix] as quoted in [IEEE92:19].

Table 5-8: Populated Proposed Framework for Model Analysis.

	Syntactic	Semantic	Pragmatic
Absolute	Size (7.1) Correctness; error-free; integrity; consistency (7.2) Modularity; structure; hierarchy (7.3)	Genericity: universality & technical independence (8.3) Completeness (domain coverage); conciseness; efficiency (8.8) Expressiveness (8.4) Similarity and overlap with other models (8.7)	Validity: authority & user acceptance (9.1) Flexibility; expandibility; portability; adaptability (9.3)
Relative	Complexity; density (7.4-7.6) Architectural style (7.7)	Perspicuity; comprehensibility; understandability; self-descriptiveness (8.5) Documentation (8.6)	Price; cost; availability (9.2) Support (9.5) Purpose; goal; relevance; appropriateness (9.6)

Similarly, there are a number of other "quality-like" evaluative concepts relating to models, which are actually composite concepts consisting of many of the criteria listed in the model and are therefore not used to populate the proposed framework cells. An example of this is usability, which includes all of the pragmatic and most of the semantic and syntactic criteria. Another is model dependability as discussed in [LAPR92 ; BARB95]. Section 5.2 illustrated in more detail how some of these composite concepts are interpreted by different authors.

The columns are separated by lines, indicative of the discreteness of the distinctions between syntax, semantics and pragmatics i.e. each criterion belongs in a single column. The absolute-relative dimension is a continuous distinction so no horizontal lines were drawn.

The various criteria are drawn from the sources presented in 5.1 and 5.2. These sources often ascribe different meanings to certain criteria and, conversely, different authors sometimes use different terms to describe a similar concept. To indicate this overlap in meaning, it was necessary to group criteria into "clusters". The clusters in map directly to the respective sections in chapters 7 to 9, where the terms are fully defined in the context of enterprise models and operationalized by means of specific measures. A separate chapter is devoted to each column, with the numbers between brackets in Table 5-8 indicating the corresponding section in this dissertation. Although it is not essential, since each criterion can be evaluated independently, generally each chapter will follow the same order as in which the criteria are listed within the framework i.e. applying the second dimension implicitly. An exception is found in chapter 8 (semantics) where the metrics to measure model similarity and completeness are derived from or based on the metrics which have been developed to measure model perspicuity and are therefore relegated to the end of the chapter. Similarly, in chapter 9, flexibility (9.3) can only be discussed after the model availability formats have been determined in section 9.2.

5.3.4 Mapping the Framework to Earlier Research

A detailed summary table has been constructed which lists all of the above criteria as found in the literature and maps them against the criteria contained in the proposed framework. The last three columns correspond to the three dimensions of the framework. Because of its size, this table has been moved to Appendix M. A few criteria are not mapped to the framework, either because they are impossible to quantify or because they are too vaguely described e.g. "aids clear thinking".

It is extremely interesting to note that [FABR98] may have come tantalizingly close to suggesting the above framework. They used the same distinction between semantics, syntax and pragmatics in the context of assessing the quality of a software architecture, but interpreted the terms quite differently and did not operationalize their approach in any practical or significant way. For the record, it must be mentioned that their relatively obscure workshop paper was only discovered after the above framework had been developed.

Intriguingly, [BRIN96, p.214] makes reference to the doctoral research of John Venable [VENA93] whereby, in his search for a theoretical basis for the CoCoA methodology, he distinguished his criteria for the evaluation of conceptual data models between those relating to semantic concepts, syntactic constructs and more generic requirements such as views and modularization.

These are by no means the only explicit references of the three categories in the context of information systems analysis: the fairly well-known publications from both Benyon [BENY90] and Stamper [STAM87] made references to these distinctions, and most computer science research in formal (programming) languages uses the distinction between syntax and semantics extensively. In particular, Stamper proposes his *semiotic framework* to classify information on six different levels: 3 on the human information level: social world, pragmatics, semantics; and 3 on the IT platform: "syntactics", empirics and physical world [STAM95] and maintains that too much of the research focuses on the "syntactic" elements.

5.4 Theoretical Validation of the Proposed Framework

Section 4.2 discusses both the conceptual and empirical validation of the proposed framework. Most of the remainder of this research, i.e. chapters 7 to 10, is devoted to the empirical validation of the

framework as well as providing (and validating) concrete measures so that each of the framework criteria can be implemented in a real-world scenario.

However, it is possible to do an initial theoretical or conceptual validation. As pointed out in 4.2.1, this validation is concerned with the foundations and internal structure of the framework from a purely theoretical perspective. Although there is no definitive list of criteria, the lists and frameworks investigated in sections 5.1 and 5.2 provide a large number of theoretical quality principles that are also applicable to frameworks themselves.

- **Construct efficiency ("Occam's razor"), simplicity and size**. The main framework dimension uses only 3 distinct yet highly natural distinctions: syntax, semantics and pragmatics. The second dimension uses a continuum with two poles: absolute to relative. This creates a very sparse matrix with only three (or, if preferred, six) cells, while providing very useful distinctions. The three distinct columns are indeed representative of the three IS references or source disciplines. The syntactic criteria and the metrics originate mostly from the exact sciences i.e. computer science and systems engineering, the semantic metrics reflect inputs from the linguistic and informational sciences, and the pragmatic measures have a distinct commercial or business focus.
- **Perspicuity**: the framework is intuitively comprehendible. It is relatively straightforward to explain the structure of the framework to a novice and there is little doubt about where new criteria have to fit.
- **Coverage and completeness**. Almost all of the criteria listed in the large number of criteria culled from the various lists and frameworks have been accommodated. Appendix M maps all of the criteria which were listed in 5.1 and 5.2 onto the framework. The framework is also "logically complete" in that there are no "gaps" between the different cells i.e. new criteria naturally fit into one of the cells, i.e. all aspects of reality are fully covered.
- **Extensibility, customisability, robustness and flexibility**. It is easy to add new criteria to, or remove non-applicable ones from the framework without impacting on the overall coherence and usability of the framework.
- **Orthogonality**: it can be argued that the two dimensions are not fully orthogonal because the more pragmatic criteria tend to be more relative in nature whereas it is much more likely that a criterion that has an absolute or objective standard is likely to be relatively syntactic in nature. However, as illustrated by the fairly even and balanced spread of criteria across the columns and rows, the degree of non-orthogonality turns out to be fairly small in practical terms.
- **Genericity, universal applicability, portability and reusability**. The framework can, under certain circumstances, easily be applied to other areas. This is considered one of its greatest strengths above all other frameworks and therefore demonstrated in sufficient detail in 10.3.
- **Formality, objectivity, absoluteness.** The framework is not formally specified and there is no strictly unambiguous set of definitions that fully express the framework. However, there does not appear to be any other formal framework in this research area. On the contrary, the framework will allow for a more formal measurement of some of its criteria (especially with respect of the semantic analysis). Also, despite the lack of formality, the level of objectivity in classifying attributes in the respective columns is high. This is not necessarily so for the distinction between absolute and relative criteria, which is itself much more subjective.
- **Maturity and authority.** Although the model is new and formulated by a relatively unrecognised researcher, it draws on authoritative literature both for the criteria as well as the theoretical grounding of the framework dimensions.

- **Theoretical foundation:** the framework is grounded firmly in semiotics as well as having a solid philosophical basis. This is also considered a very specific strength of the proposed framework.
- **Modularity and hierarchy.** The framework has only two dimensions, of which one will not be explored in this research, so it has a relatively thin structure.

It will be noted that many of the criteria listed above also feature as *enterprise model* evaluation criteria within the proposed framework. This leads naturally to the question whether the framework can also be used in a self-referential manner to validate itself. That this is indeed the case will be followed up in section 10.3 in the context of the generalizability of the framework.

The arguments above address also the general problems experienced with the frameworks imported from other disciplines (as raised in 5.3.1), namely the need for a theoretical foundation and specificity (to enterprise modeling) as well as generality (extendible to other areas).

Overall, it can be concluded that from a conceptual or theoretical point of view, the framework appears to have a fairly high validity. The second dimension (absolute ⇔ relative) appears to be somewhat less valid and, as discussed higher, will not be followed up in explicitly in the remainder of this research, although it is a prime candidate for future research.

In order to provide the probably more significant empirical validation of the framework, it is necessary to construct a test bed of enterprise models against which to operationalise and test the framework and its criteria, as explained in chapter 4. The following chapter, therefore, presents a survey of enterprise models and details which ones have been selected for this empirical validation process.

Chapter 6: A Survey of Generic Enterprise Models

Having developed the overall analysis framework, it is now time to build the model database on which to test and validate the framework.

This chapter provides an overview of all the generic enterprise models which could be found in publicly available sources and met the model selection criteria as set out in the methodology chapter. At the end of this chapter, details of some other generic enterprise models which did not meet the criteria have been added.

Although almost all of the models represent *generic enterprise* models, a number of "boundary cases" were included for the explicit purpose of validation of the various analysis techniques. The following models can be considered as marginal or boundary cases:

- BelgAcc is formulated in a different language (Dutch) and has a very sparse relationship space.
- Miller is a small model originating from a very different reference discipline (systems theory) and based on a fundamentally different paradigm, using a non-standard terminology.
- AKMA is a standard, high-quality data model but formulated for one specific, vertical industry namely the health care industry.
- Random and semi-random were specifically constructed (see Chapter 4) as syntactically "seemingly correct" models without or little semantic meaning.

Also note that some possible enterprise models are missing from the database because of the difficulty of finding sufficiently large, explicit models. In particular, no suitable enterprise model from economic theory such as neo-classic micro-economics or Marxist theory or a possible network model of the organization stemming from a sociological perspective was found.

Although the models below are not listed in a strict order, the rough order used for reference disciplines in Chapter 4 has been adopted for this listing. Hence the "systems engineering" and purer "IS models" appear towards the back. It remains important to remember that these models originate from a wide field of reference disciplines or research areas. The specific roots of a model has a major impact on the scope and perspectives which have been adopted in the model and the reader is advised to refer to chapter 3 where these biases or approaches are explained in more detail.

Very important for the purposes of the verification of this research are the capturing notes, i.e. the various problems and issues that were encountered in the model capturing and conversion process. Due to the lengthy nature of many of these notes, they have been moved to Appendix A of the thesis. This Appendix also contains typical examples of the original model source i.e. text fragments or sample diagrams. They are annotated with comments on how a typical conversion to the database meta-model was implemented.

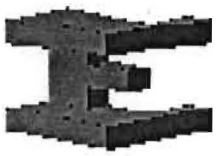
6.1 Overview of Models Included in the Data Base

6.1.1 AIAI's Enterprise Ontology

The Enterprise Ontology is a collection of terms and definitions relevant to business enterprises. It was developed as part of the Enterprise Project, a collaborative effort to provide a framework (including a method and computer tools) for enterprise modelling. The idea was to provide one set of terms and definitions which adequately and accurately covers the

relevant concepts in the enterprise modelling domain. Its main purpose was to facilitate communication between different parties, including users, developers and computer systems. The Enterprise Ontology was completed in 1996 in a natural language format and subsequently formalized, with minor modifications, in Ontolingua in 1998. [USCH98]

6.1.2 EIL's TOVE Ontologies



The goal of the TOVE (TOronto Virtual Enterprise) project is to create a data model that aims to provide a shared terminology for the enterprise, defining each term in a precisely using a set of axioms that will enable computer-based deduction of the answer to many "common sense" questions about the enterprise. The TOVE ontology is an ongoing research project; typically each ontology is a doctoral output. To date, not all sub-ontologies have been completed e.g. the electro-mechanical product, information resource, quality, service ontologies are still outstanding. The final version will comprise 19 separate ontologies. Some of the ontologies were subsequently included in the PSL specification. [FOX93b; FADE94; FOX95; KIM95; LIN96].

6.1.3 CYC Upper Ontology

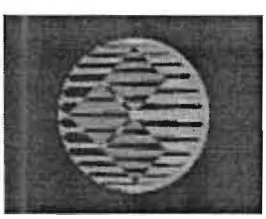
The complete CYC ontology is a multi-year project to attempt to formalize common sense. It has been under development since 1984 by AI pioneer Doug Lenat. The knowledge base is built on a core of over 1,000,000 hand-entered assertions (or "rules") designed to capture a large portion of what is normally considered consensus knowledge about the world. It is claimed to be the world's largest and most complete formal knowledge base; CyCorp also developed a common-sense reasoning engine to accompany the knowledge base. In 1997, CYC released the beta version of the Upper CYC Ontology (version 2.1) i.e. the uppermost 2691 concepts (also called CYC constants, terms or units). A new, fully updated "open source" version, with approximately 6000 concepts, was scheduled to be released in the second half of 2001 but this only happened in December 2002. The "generic enterprise model" has been distilled from the old version by selecting a sub-set of 777 CYC constants which was deemed to be related to the enterprise domain. Although the selection process is subjective, including or deleting an extra few hundred constants does not really change the nature of the model, just its relative size.

6.1.4 ARRI's Small Integrated Manufacturing Enterprise (SIME) Model

ARRI

The goal of ARRI's Enterprise Engineering program is to research and develop methods and tools to implement the integrated enterprise. It is developing enterprise reference models to provide a standard understanding of the manufacturing enterprise, using the IDEF0 methodology. It has developed an enterprise model for the operations of a Small Integrated Manufacturing Enterprise (SIME, published 21 December 1990). These form part of a set of related models, including one for continuous enterprise improvement, enterprise transformation and enterprise performance management. The SIME operational activity is broken down into 6 sub-activities, each of which is in turn broken down into approximately 4 sub-sub-activities.

6.1.5 The Purdue Reference Model for Computer-Integrated Manufacturing



The Purdue Reference Model is one of the earliest attempts to create a standard reference architecture within IS, and can be seen as one of the origins of the

enterprise engineering discipline. The initial idea was to check which and to what extent each operation or function within a manufacturing organization was automatable. The Workshop Committee created this model, using the now dated data flow and hierarchical modelling notations. It was hoped that the model would become the foundation of a standard reference framework within the CIM community. The model was subsequently used in the standardization work of the Working Group 1 of subcommittee 5 of Technical Committee 184 (Industrial Automation Systems) of the ISO and is the background for the PERA (Purdue Enterprise Reference Architecture).

The Purdue model consists of a set of data flow diagrams, annotated with a detailed lexicon of terms (for the "data dictionary"), and a scheduling and control hierarchy mapped to the diagrams [WILL91]. The Purdue Reference Model document also includes many additional inputs, such as the context of the model, the process of implementation, software requirements for CIM, the specification of interfaces and the role of the human in the automated CIM environments. Although these make up a significant portion of the publication, they are not relevant for the model capture.

6.1.6 The Nippon Steel Corporation Industrial Automation System Model

The Purdue Reference Model document contains an Appendix V which describes "an adaptation of a recent Japanese Industrial Automation System Model (dated June 11, 1987, Anonymous)". Although the same notation is used as for the Purdue model, the underlying philosophy is very different and the resulting data-flow model, apart from being far less detailed, is "much different from than the Purdue Reference Model [WILL91]. Since the capture involved only little extra effort, it was also included in the model database.

6.1.7 The Belgium Accounting Framework

The accounting model is the oldest known general model that has been developed as the basis of a business information system. Not only are the oldest surviving writings, Mesopotamian clay tablets, assumed to be records of business transactions; but the "double entry" accounting model in its most basic form stems from Pacioli [1494]. National accounting bodies in most countries have adopted and/or expanded the basic model in specific ways, usually by means of specific government legislation or through the publication of standards and guidelines by the national accounting authority.

The most detailed models have been developed in Western Europe, perhaps because of an arguably more pronounced culture of statutory public accountability. In 1972, the Commission of the European Community promulgated its 4th directive promulgated to provided a role model for harmonizing the various national accounting standards. Belgium was the first country to translate the directive into specific legislation by means of a comprehensive set of laws on "*de boekhouding en de jaarrekening van de ondernemingen*" dating from 17-Jul-1975, 30-Mar-1976, 24-Mar-1978, 1-Jul-1983, 12-Jul-1989, 6-Aug-1993, 6-Apr-1995 as amended/supplemented by the Royal Decrees of 15-Dec-1978, 16-Jan-1986, 30-Dec-1991 and 27-Apr-1995. These laws provide an extensive compulsory chart of accounts and set of financial statements to be used/completed by all mid-sized (which can use a subset) and large companies in Belgium. For instance, in 1992, 14 417 (large) companies had to submit a full set

of accounts and 139 572 (medium-sized) companies an abbreviated set, to the Financial Statements Centre which is run under auspices of the National Bank of Belgium [REYN94].

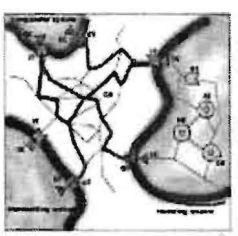
Typical accounting transactions, exemplifying typical "relationships" in the accounting model, were captured from one of the leading accounting textbooks [REYN94].

6.1.8 The U.S.B. Growth Model

The U.S.B. Growth Model is a menu-driven financial spreadsheet for use with Lotus 1-2-3. It projects pro-forma financial statements, cash flow situation and performance measures of growth businesses for a five year period on a monthly basis. It has been designed to provide maximum flexibility although a conscious effort was made to reduce the input requirements to an absolute minimum. The user can manipulate 27 to 72 variables and 10 parameters (not including the required historical data input). Sensitivity and target analysis capabilities are provided in addition to the extensive printed and graphical reports.

It was developed as part of the requirements for the M.B.A. degree at the Graduate School of Business at the University of Stellenbosch. Although relatively small, it was pushing the limits of the personal computer hardware (RAM requirements) and software (Lotus 1-2-3) at the time of development (1988). Due to the generally prevalent 640K RAM constraint on PCs, version 2.1 was re-written from a Lotus v.2 to Lotus v.1A format and released locally (in South Africa) as freeware. Although the model was meant to be a generic financial model for small and medium-sized businesses, a specific focus was to allow for 1 to 5 year forecasts of the maximum growth rate a business could sustain given certain financial decisions (e.g. profitability, financing mix, dividend policy). Despite its small size, it is considered to be a good example of generic financial models [VANB88]

6.1.9 J.G. Miller's General Living Systems Model



In 1978, James Grier Miller published his "Living Systems" book, a densely printed 1100-page tome. In it, he develops a general model of living systems, consisting of 19 subsystems (later adding *timer* as a 20th subsystem) and states a large number of hypotheses relating to the functioning of living systems. Particularly innovative for the time is the attention he gives to the information processing function, which accounts for half of the subsystems. He subsequently checks his hypotheses against a variety of living systems. His hierarchy of living systems stretches from the individual cell, via organs, organisms, individuals, societies etc. all the way up to the supranational system. Well over 100 pages are devoted to the organization as a particular type of living system. A lot of the research is now somewhat dated, but his approach still rings very novel and original and it appears to be a pity that it has not been adopted more seriously in IS and other disciplines. However, after two decades, at least one IS-related research project is directly based on his model: [COFF97], although there would be significant scope for applying his theory to user-interfaces and communications theory. Although it is by far the smallest of the models surveyed here, it has been included specifically because of its originality and extreme generality [MILL78].

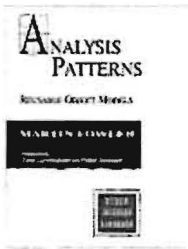
6.1.10 Bill Inmon's High and Mid-level Data Models

On his website, Bill Inmon, "the prophet of Data Warehousing", provides two types of generic data models: industry-specific generic data models and functional generic data models. The industry

specific generic data model reflects the basic business activities of a company engaging in commerce. The industry specific generic data models provided are: airline, banking, insurance, oil/gas, railroad and university. The functional generic data models reflect common functions done in any company and consist of the accounting, marketing, sales, corporate tax and contracts generic data model. The "industry-specific" manufacturing data model was also included as a generic data model, in line with the ERP models. An organization that wishes to produce a complete model of the entire corporate environment could select several functional data models and combine them with an industry specific data model. The result would be a comprehensive data model of the entire corporation: "The business similarities between companies in the same industry are much more striking than the differences. As such, the data models that represent those businesses are likewise very similar."

The generic data models available on his website consist of high and mid-level generic data models. The mid-level diagrams contain important or typical attributes. The models appear to be based on his wide experience with implementing data warehouses across many industries. Although he is a co-author of the *data model resource book* [SILV97], there seems to be little overlap between the models.

6.1.11 Fowler's Object-Oriented Analysis Patterns



martin fowler .com

The book contains a large number of object-oriented analysis patterns, based on Fowler's consulting experience where he saw many problems (and their solutions) repeat innumerable times. The book is probably the best known work on OO analysis patterns. Although it was published in 1997, much of the conceptual work was done earlier and the models are therefore presented in Odell's notation (an EERD type notation) instead of UML diagrams.

The majority of the analysis patterns are very high level, although quite a few chapters contain technical implementation or design-related patterns. [FOWL97].

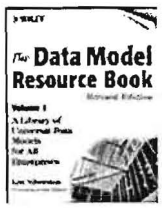
6.1.12 Hay's Data Model Patterns

Hay's book "Data Model Patterns" describes a set of standard data models that can be applied to standard business situations. Although his models are described as patterns, and despite the fact that many of his models are indeed at a more abstract level than most data models, the models are not quite at the same high level as Fowler's patterns.

DATA MODEL PATTERNS
Data Architecture in a Box™

The book contains a large number of very well laid-out, clear and detailed diagrams with full documentation on the reasoning behind the models. There are well over 150 diagrams and tables. The model diagrams show relationship cardinalities and relationship role names. The book and its models are of such quality that they can easily serve as a showcase or benchmark for enterprise data modelling. Hay has since updated the book with some more advanced (higher level) patterns in [HAY98a]. These were not included because they overlap partially with some of the models in [HAY96].

6.1.13 Silverston et al's Universal Data Models



[SILV97; SILV01] provide a large library of common data models and data warehouse designs for common functions, as well as the rationale behind many of the base 'universal data model' constructs: "Let's face it, most data models are made up of common constructs that have been developed countless times before in other organizations." The book consists of a comprehensive set of detailed models along with instructions to convert the logical data models into enterprise-wide data warehouses and data marts (the "Inmon" influence).

The first edition was published in 1997. An update (mainly the addition of a chapter) that appeared in 2001 as "Volume 1" provides common data models and data warehouse designs that are useful across industries. Volume 2 provides additional data models applicable to specific industries. Since 2001 edition became available only after the model had been captured already, the 1997 edition is used in this research.

6.1.14 AKMA's Generic DataFrame

AKMA's Generic DataFrame© is a generic model that exploits the fact that the main information needs of a business in a given sector, or even across sectors, are very similar. The DataFrame is intended as both an operational model and an informational model. It is a formal model generated using the Popkin Software CASE/modelling tool Systems Architecture 2001. It is very generic which means that it can be applied easily in many situations and "objects" may be reused throughout the model. It makes fairly heavy use of inheritance structures.

The high-level conceptual model that outlines the key concepts and philosophy of the DataFrame approach is publicly available from the AKMA website. It contains those high-level objects that relate to major subject areas in the full model as well as the first layer of objects from the Generic Model to allow prospective clients to get a feel for what the full DataFrame provides.

6.1.15 NHS Generic HCM Class Model

The models on the NHS web site represent their understanding of the provision of patient care in the English and Welsh National Health Service. They are information models and their purpose is to describe a healthcare system.

One of the models (Provide Patient Care) reflects the real world processes carried out in a healthcare system. This includes the delivery of care to patients, the management of the delivery of care and associated resources, and links to the underlying external knowledge, terminology and classification systems. The Generic Class model describes the types of data (or objects) which underpin the Provide Patient Care model, and a lot more besides. Only the Generic Class model was used for this research. Its advantage is that the domain appears generic enough yet it is a good example of a more industry specific model.

6.1.16 Marshall's BOMA Model

BOMA™

Chris Marshall describes a specific way in which to model enterprises from a business rather than a technical perspective. His book [MARS00] contains a number of high-level, conceptual models, very much at the level of Fowler's analysis patterns, but from a proper object-oriented perspective. His meta-model is based on the key business object types: entity, process, purpose / organization as endorsed by the OMG. It is not a coincidence that he is closely associated with its Business Object domain task force.

The book not only includes the diagrams and a very readable description of / motivation for the models, but it also includes an appendix in which his models are applied to a case study. In addition, the BOMA CD-ROM included with the book provides an electronic copy of the diagrams as well as the necessary JAVA code to generate and/or customize the classes [MARS00].

6.1.17 San Francisco Application Business Components

IBM

The San Francisco Framework is a commercial, OO application framework. It consists of a set of generic, high-level JAVA classes representing business entities. These classes cooperate in a user-defined way to implement core business processes. It is one of the first examples of a framework to include high-level business functionality, instead of the usual low-level system focus. The framework's goal was to prove that flexible yet robust business components can be built for multi-platform systems with a minimum of complexity, expense and time investment. The underlying principles of SanFrancisco are those of object-orientation, portability, client/server technology, platform independence, sound architectural principles and RAD [BENN00]. IBM is now no longer marketing IBM SanFrancisco, since it has been superceded by WebSphere Business Components Studio as from 11 Sep. 2001.

6.1.18 SAP R/3's Reference Model

SAP

SAP R/3 is by far the most successful ERP system on the market and the perfect example of how a generic model can apply to organizations across a multitude of industries; although the extensive customization and implementation processes illustrate equally well the gap that remains. The SAP R/3 Reference Model is based on the research done by Prof Scheer. In [SCHE98], he published the underlying "Reference Models for Industrial Enterprises". This is an extensively researched and well-documented 770 page "model" for general enterprise models. The ground work for the models was published in a first edition in 1989. This, in turn, was based on the research done as part of the "Cologne Integration Model" (KIM).

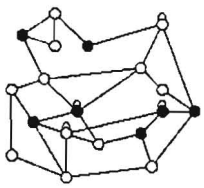
The book contains 580 figures but, luckily, there is a much smaller number of "summary diagrams". Although the book contains both process as well as data models, only the latter have been captured (as explained in the methodology section). The model was captured from the 24 summary data diagrams. It must be noted that the SAP ERP system relies heavily on the process models; also its underlying data model has developed quite substantially from its roots, not in the least due to its move to object-orientation and e-commerce integration.

6.1.19 The Baan IV DEM Business Reference Model

Baan Company is one of the leading ERP solution providers worldwide. Its current product is the Baan release IV system. The Baan system is based on a 1985 software package that included finance, manufacturing and distribution modules. This package was enhanced, using an MRP II approach, in the 1989 "Triton Software", which was really the first release of the ERP system. In 1990, the system was moved to a client/server environment and during 1993 and 1994 the process, transportation, project control and EIS modules were added.

The reference model underlying the system has, as far as known, not been made available in any publicly available document, although the model can be accessed easily by any license holder through the Orgware Dynamic Enterprise Modeler (DEM) tool. Not only would this probably infringe intellectual property rights, but it also clashes with the methodological objectives of public availability in order to ensure repeatability and verification by other researchers. Hence a decision was taken to "reverse-engineer" the model underlying Baan IV by using a publicly available book that describes its structure in detail. After some research, it was clear that the tables and screenshots in [PERR98] were appropriate and the book covered a sufficiently large area of the Baan IV system.

6.1.20 The Random, Semi-Random, Ottawa-Big and Ottawa Dense models



Ottawa Business Journal
We mean business.

These models are fully described in sections 4.3.3 to 4.3.6. To summarize, Random is intended to be a fully random model i.e. a model without semantic content. Semi-Random is intended to be a semi-random model, with entities related to the domain (the organization) but with meaningless relationships between the entities. The Ottawa-models are a semantically based "hyperlink" version of a business terms dictionary i.e. a "model" consisting of domain-specific concepts with relationships representing semantic references in the lexicon definitions of the concepts.

6.2 Some Enterprise Models that Did Not Meet the Criteria

During the model survey effort, a number of other enterprise models were found that did not meet the criteria for inclusion into the database. They are listed below, in no particular order, along with a short description as well as the reason for their exclusion.

6.2.1 Presley's Holonic Enterprise Modeling Ontology

In his doctoral thesis, Presley develops a "holonic" enterprise modelling ontology in an attempt to combine some of the principles upon which the TOVE and AIAI ontologies are based with a holonic systems dynamics approach to enterprise modeling. There are also some additional architectural inputs such as those from CIMOSA, ARRI and PIF.The resultant model is fully described in Appendix B of his thesis: [PRES97b:241-273]. The model has not been included in the database because most of the concepts are too high-level (abstract). In effect it is a well-structured meso-model that is not specific enough to generate an enterprise model. It would be ideal as the basis of an intelligent enterprise modelling tool.

6.2.2 J.D. Edward's OneWorld

J.D. Edwards has developed a popular ERP package under the name "OneWorld". It competes in the same market as Baan and SAP and the underlying model therefore covers a similar domain. The technology on which the system is implemented is accordingly also very similar. There are a number of books describing the package, of which "*J.D. Edwards, OneWorld: The Complete Reference*" [MILL01] is the only one containing a detailed description of the package itself. In Appendix A of that book, a full listing of all the tables used by the system is provided. The listing contains, *inter alia,* the 1612 "Business Data Tables". "These tables represent all tables specific to actual business data, including the address book, accounts receivable, accounts payable, inventory, sales orders, forecasting, manufacturing, distribution, human resources, payroll, purchase orders, and many, many more." (The other types of tables are system / implementation specific e.g. control tables, central objects, local tables etc.).

The full listing of tables was captured in the research database since each table can be thought of as corresponding to a model entity (refer to the reengineering procedure for Baan). However, unlike for Baan, no indication of relationships or generalization structures could be inferred, leading to a relatively "sparse" model, despite its great number of entities. The lack of relationship data was sufficient ground to discard the model from the sample, since very little of the syntactic analysis would apply.

6.2.3 Other ERP Models and related standards

Apart from Baan, OneWorld and SAP, there are a number of other ERP solutions on the market. Although supporting books exist for many of them (especially the PeopleSoft package), none of the books currently on the market seems to contain a detailed enough description of the package to enable a reverse-engineering effort as was done for Baan.

An interesting document in this respect is the "V-ERP White Paper/Roadmap" produced by the "V-ERP working group of the OMG MFG DTF. This draft document (v.0.3 , released 29-July-1997) is an attempt to provide a roadmap for RPFs in the ERP domain and to provide generic process maps to allow different ERP packages to inter-cooperate. Although the document contains a number of sample diagrams (originating from Texas Instruments), this effort appeared to have stalled and was subsequently superceded by the MDC approach.

A related standardization attempt is the "*Business Engineering Model*" which forms part of the *Open Information Model* produced by the Meta Data Coalition (MDC). The Open Information Model also aims to provide standards for data types so that modelling tools could exchange modelling information. One of the standardization efforts is the Business Engineering Model (a draft was produced in 1999) to look at meta types for business engineering.

It is interesting to note that in 2000, the OMG and MDC agreed to consolidate their efforts and work together.

6.2.4 The db Trader Enterprise Architecture Data Model

At http://www.isy.vcu.edu/~paiken/projects/dbna/kw/contents/framework/dbtraderarch/ an overview of the dbTrader Enterprise Architecture Data Model is given, with a mapping

onto the Zachman framework. Details of the framework are available from the same website, and diagrams for the following business subject areas are already available: account, accounting & evaluation, corporate action, derivatives, foreign exchange, order, organization, position, security, settlement & stock record and trade.

The model has not been included in the database for the following two reasons:

- Firstly, it is still incomplete and omits too many critical business areas.
- Secondly, it is too focused on a vertical market namely the trading of securities.

6.2.5 Public Petroleum Data Model

The Public Petroleum Data Model has been developed by an independent, not-for-profit association representing more than 100 oil and gas companies, vendors and regulatory agencies. The PPDM is intended as a platform and vendor-independent standard data model for the petroleum industry. It is felt that standardizing the data entities across the industry would leverage the information assets within the industry and facilitate cooperation and trade, reducing the transaction costs and potential data mismatches. The current release is version 3.6.

The reasons for not including the model are the following:

- The model is very focused on a vertical market, namely the petroleum industry. As such most of it is concerned with very specific processes and entities e.g. those relating to seismic data, wells or land management information.
- The model can be downloaded only by members of the PPDM Association. Although anyone from the public can become a member, the fees are too high for individual researchers.

6.2.6 Epicentre Logical Data Model by Petroleum Open Software Corporation

A "competing" not-for-profit consortium in the petroleum industry, the Petroleum Open Software Corporation (POSC, http://www.posc.org), has also developed a logical data model, called Epicentre, to facilitate data sharing between the information systems of its partners.

Although the model is supposed to be available only to members, the full schema with all entities, its attributes and the named data types, is available on http://www.august.com/epicentre/. Despite its public availability, the model is not included in the database mainly because of its very vertical focus (most entities do not relate at all to the generic enterprise).

6.2.7 The NORNE Enterprise Model

A high-level process model for NORNE (a Norwegian exploration effort) is available from http://www.pakt.unit.no/~statoil/norne/ . It is too small, and neither specific nor generic enough to qualify for the database.

6.2.8 The Visible Universal Model

Visible Systems Corporation has developed its "Visible Universal Model", marketed as a "Universal Enterprise Information Architecture". It proposes to fast-track the

development of high-quality data warehouses, executive information systems and decision support systems. It comes with a ready-made template to build "universal business objects" and processes. It reputedly contains over 300 universal business processes. The Data Model contains more than 50 business subject areas (the "primary business objects"), that are in turn subdivided into over 600 entities with 1000+ attributes. More information about the model can be found at http://www.ozemail.com/~visible/papers/UNIVMOD.html and http://www.ozemail.com/~ieinfo/universal.htm. The model has not been included in the database due to its cost ($100,000). A request for a research inspection copy was refused.

6.2.9 The ADRM Data Models

ADRM (Applied Data Resource Management) provides a large set of industry-specific data models. Although they all revolve around a common core set of data entities, they are customized and enhanced to support the specific needs of 7 industries, 27 lines of business and 30 business areas. Examples of supported businesses are food & beverages, chemical, credit unions, retail banking, high-tech components, wireless and other industries. The models consist of three levels:

- A high-level integrated enterprise model.
- A more detailed model suitable for data warehouse development.
- Logical models for specific business subject areas.

Typical model sizes are 300 to 500 tables for the enterprise and data warehouse model (with 1500 to 2500 attributes/columns) and 3000 to 5000 tables (10000 to 15000 columns) for the logical models. A more detailed description of the models can be found on ADRM's website http://www.adrm.com . The models have not been included in the database due to their high cost. A repeated request for a research inspection copy was never acknowledged.

6.2.10 IBM's WebSphere Business Components

IBM released a beta version of its set of WebSphere Business Components (WSBC) as part of its support for its VisualAge for Java product suite. WSBC is available from http://www-4.ibm.com/softare/webservers/components/download.html . A more detailed inspection reveals that this is a framework specifically targeted at RAD for e-commerce enabled business. The framework has not been included for the following reasons:

- There is substantial overlap with its SanFrancisco Framework.
- A lot of the components are e-commerce specific.
- The set was still very much under development when the model analysis began ("these components are so cutting edge that they have not even made it into the WebSphere Business Components Studio product yet").

6.2.11 ebXML Business Process Project Team

This project team (see http://www.ebxml.org/project_teams/business_process/) is charged with developing a standard for full inter-organizational process integration. Its work is partly derived from the efforts of the OAGIS (http://www.oagis.org) and the UN/CEFACT Electronic Business

Transition Working Group (http://www.ebtwg.org/).

As part of this effort, they have prepared some proposed standard high-level objects e.g. agreement, business document, business event, business process, market, party, party type etc. The current model is too small (too few entities) and too transitory to be incorporated in the database.

6.2.12 The Wizdom Manufacturing Enterprise Reference Model

Wizdom Systems has condensed its business process re-engineering experience in a book and accompanying CD-ROM containing a generic Manufacturing Enterprise Reference Model, represented by 30 IDEF0 process models and associated narrative text. The model costs US$495 and was outside the budget for this research. A request for a research inspection copy was not acknowledged. No cheaper second-hand copy of the book could be located on Amazon.com.

6.2.13 The Engenia Organization Model

Engenia describes its *Organization Model for Learning Organizations*as a comprehensive, well-structured classification model for the most important information about an organization. It draws on both management theory and practice but focuses on an analytical and conceptual approach to strategy planning, change management, BPR/BPI, knowledge management as well as developing software applications and preparing for data warehouse projects. The website http://www.ingenia-ltd.demon.co.uk/OrganizationModel/OrgModel.htm provides no detailed information on what is claimed to be the fourth version of the model (1994-1998) but apparently a book is under preparation which will contain some of the model details.

6.2.14 Miscellaneous Tiny Enterprise Models

The following models are also enterprise models but contain too few entities to be included in this research:

- **Decision Dynamics' Models.** As part of its efforts to promote its system dynamics modelling methodology, Decision Dynamics has developed some high-level generic flow models for the enterprise. The models are available from its website http://www.decisiondynamics.com/.
- **IS/Modeler's Enterprise Model**. A very simplistic enterprise model of core business processes can be found at its website http://www.ismodeler.com. It contains too few entities for consideration.
- **BOT*i*CMAP *QuickModel***. Only the high-level model is available for public download, intended as an illustration of its tools and methodology. It can be found at http://www.botic.demon.co.uk/quickmodel.htm .
- **OutSights' Models**. Gene Bellinger, OutSights, offers a number of disjoint enterprise model diagrams on his website http://www.outsights.com/systems. The primary purpose of the model fragments is to illustrate "The Way" which is Outsights' implementation of systems dynamics modelling. The models are developed using its proprietary modelling and simulation tool "I-think".

- **Hsu et al's Core Information Model.** In their article [HSU94], the authors propose a generic model to support CIM, consisting of 6 sub-models. Although their model is formal and quite rich in terms of semantic content, it contains too few entities and relationships.
- **Stewart's "Enterprise-in-a-Box".** In his article [STEW01], Jim Stewart presents some high-level enterprise data model elements in support of his case for the value of an enterprise data model as part of the IT architecture.

6.3 Using XML as the platform to share the models with other researchers.

Within the limitations of copyright legislation, the database with captured models will be made available to other researchers. The question arises as to the best format or platform in which the models should be made available. It should be noted that this refers to the ***external format*** which is used for sharing the models with other researchers. This is different to the ***internal formats*** which were used for the actual research, since this depends on the research applications' input formats e.g. flat ASCII files for semantic analysis, spreadsheet .xls format for calculating metrics etc. After careful consideration, it was decided that the only real alternative to consider was XML (Extensible Markup Language).

6.3.1 Why use XML?

The following are the main reasons why XML is ideally suited for the exchange of the model database:

- XML is a platform-independent, non-proprietary, internet-oriented way of sharing a textual database. The type of hardware or operating system used is immaterial.
- Interest and support for XML is growing exponentially. It is fast becoming the de-facto standard for sharing data between heterogeneous applications and between different organizations.
- XML provides extreme flexibility in organizing data yet the XML data can be manipulated relatively easily, e.g. by using XLST.
- The fact that it is a text-based mark-up language, like HTML, makes it human-readable in principle, although it is primarily intended for machine (application) interpretation.
- Its structure and design makes it ideal for *hierarchically* structured data, which fits the meta-model of the database perfectly.
- It is relatively easy to transform the data into any desired output format e.g. in flat or formatted text or even graphics. The data can be formatted in an attractive on-screen format using the power of XML Style Sheets (using XSL - XML Style Language) or in any "pretty-print" file format using an XLST. It is also possible to do limited processing e.g. data selection using XQL (XML Query Language).

6.3.2 The different XML options

Choosing XML is only a first step. In a way, the choice of XML is akin to saying that one chooses the ".MDB" format (the database format used by Microsoft Access). There is still the important and often tedious step of defining the actual database structure i.e. the tables, their various fields, the meaning of the fields and the relationships between tables. This is done by specifying the tags and the relationships between them i.e. the definition of the XML data elements. A number of options are

available to define the allowable data tags or schema to describe the vocabulary and the structure of the XML data. The procedure for converting the model database into an XML database is based on the process as suggested in [FONG01].

The original and oldest method is by using a DTD (Data Type Definition) schema. Although this can be embedded in the document, the preferred method is the construction of a separate, external .DTD file. There are a number of drawbacks to using DTDs (see [MØLL01]), the main one for our purposes is the lack of (attribute) inheritance support in the hierarchical structure (as per meta-model). There are a number of alternatives, the most important being the XML Schema, with the SOX (Schema for Object-Oriented XML) specification of XML Schema probably the most suitable contender in the context [DAVI99]. However, XML Schema do still lack popular tool support, are more complex and await final standardization.

For the actual generation of the DTD or schema for the model database, the following approaches presented themselves.

Approach 1: The use of XMI (XML Metadata Interchange format).

XMI is an XML standard proposed by OMG in 1999 with the main purpose of supporting the interchange of model data between modelling tools. Support for XMI is growing fairly rapidly with most new versions of modelling and CASE tools supporting XMI import/export functionality. The following arguments can be considered in respect of XMI:

- XMI has been specifically proposed for the purpose of model interchange.
- It has a large share of "mind-share" i.e. it is well-received by the tool developers.
- There is a natural support for model diagramming especially UML diagrams.
- There is considerable tool support including a number of public domain and freeware modelling tools (e.g. Argo/UML).
- On the negative side, XMI is quite complex, and not really human-readable: "so far most of the interest in XMI has come from tool vendors and has inevitably been for rather heavyweight tool integration" [STEV01]. Although this is not a problem when standard modelling tools are used, it becomes problematic when researchers make use of custom developed analysis software. This does not support XMI and the desired data elements can be buried under loads of meta-data elements.
- There is still some degree of incompatibility between the different tools' interpretation of XMI. To illustrate, consider the statement that "MagicDraw UML 4.5 can now read and write *UNISYS UML XMI*" (XMI Watch – Objects by Design – http://forums.objectsbydesign.com). In the same vein, someone else reports on the same forum that: "Try XMI export in Rational's Rose, then import that same XMI export back into ROSE – does it work? You will find that it probably won't!"
- XMI is really geared for exchanging actual models between tools, independent of notation. It is *not* geared towards meta-models i.e. one single file containing multiple alternative models of the same domain.
- Most research analysis tools do *not* support XMI. None of the non-modelling based tools used for semantic analysis and statistical analysis, for example, recognize native XMI. None of the mainstream (standard) database support XMI directly. A particularly important case was the fact that the Archi knowledge management tool supports XML import but is likely to choke on XMI.

- Where XMI is absolutely essential, a conversion from XML to XMI should always prove to be relatively straightforward though not necessarily a trivial exercise using a XLST.

To illustrate some of the complexity of XMI, Appendix G lists the minimal required tags to be specified in the XMI schema.

Approach 2: Use of OIL or DAML: an XML standard for ontologies.

OIL (variously defined as Ontology Inference Layer and Ontology Interchange Language) is an XML standard specifically developed for the sharing of ontologies but aimed at providing inference logic as well. The language has been specifically designed to support the standard modelling primitives and provide "simple, clean, and well defined semantics" [BECH00].

What distinguishes it from XMI is the fact that, apart from a "machine-readable" set of definitions, available as an XML DTD, XSD AND RDF schema definition, there is also a formal definition of a pseudo-syntax which includes formatting such as bolding and indentation in order to enhance human readability. The full specifications are available from http://www.ontoknowledge.org/oil/.

If only the exchange of the static aspects of ontologies is required, the XOL - XML-based Ontology (exchange) Language - is simpler and more straightforward. Alternatively, derived from OIL is DAML (DARPA Agent Markup Language), an ontology markup language which is also easier to use. Its specification is available from http://www.daml.org.

The only real problem associated with the OIL and DAML tags is that they do not map cleanly and intuitively to the meta-model, since they use the terminology common to ontology researchers. Modellers are unlikely to relate well to many of the terms e.g. a "slot" instead of an "attribute".

Approach 3: Tool-generated XML.

Given the recent increase in support for XML as a data interchange format, it is relatively straightforward to make the model database available in "machine-generated XML". Two alternatives can be considered here.

The natural choice is to use the MS-Access XP "export-to-XML" feature. However, although the documentation states that "related tables can also be exported to the same XML file", this feature is not accessible from the current user interface implementation which exports only one table at a time (although the technical Microsoft documentation appears to indicate that it is implemented in the underlying source code). It appears that even the full implementation would not export the entire database. Attempts to export the entire database in one single XML file proved unsuccessful.

An alternative is the use of external XML tools. An example is DB2XML, a JAVA-tool available for transforming data from relational databases and release by the Informatics department of the University of Wiesbaden. The database is accessed via a JDBC driver and the tool was tested against a wide variety of databases (and drivers), including Oracle, DB2, MySQL and MS-Access. Although DB2XML does support a full database transform, it does not take into account the hierarchical structure as implied by the relationships between the tables. Apparently this is due to the fact that "simple databases such [as] MS-Access do not support schemata" [TURA02]. Although the tool worked fine for any one table, the hierarchical structure as implied in the meta-model was again lost.

The following are the advantages of tool-generated XML:

- Easy to generate: all it takes is a couple of clicks.
- Much better portability than XMI: most end-user applications will support generic XML and for those that do not, a relatively easy CSS-based conversion process can format the data into the desired format.

- Fairly good human readability: unlike the bloated "HTML"-files generated by, say, MS-Office (check e.g. the HTML version of an MS-Excel table), the "XML" files from both MS-Access and DB2XML are relatively sparse and easy to read.

The main drawback is the fact that the apparent hierarchical structure of the meta-model is not reflected in the DTD.

Approach 4: Handcrafted XML

In the end, it was decided to create a DTD and XSD by hand. The main arguments for this approach are the guaranteed human readability (minimal clutter), a "white box" understanding of the DTD and its universality. The only problems with handcrafted XML are the extra work involved as well as the need to explicitly test the generated files for well-formedness.

6.4 The EnterpriseModels XML Database

The table mapping the meta-model to the XML tags that were used for the EnterpriseModels database, can be found in Appendix H. A total of 37 different XML tags were used. Note that a few minor deviations from the meta-model were introduced for practical or completeness reasons. In particular it is worth noting the following:

- The <ModelLogo> element was left as a null value. Although a standard exists to incorporate graphics files in an XML database, this results in long strings of meaningless alphanumeric characters and may not be correctly interpreted by some common first-generation XML parsers.
- A very few models included descriptions for their domain relationships and/or groupers, so additional elements were introduced for this.
- In the tables in chapters 7 to 10, the ModelID is often abbreviated to a two-letter ModelCode. This was added to the database as the ModelKey.
- For the semantic analysis, it was necessary to parse or convert many of the entity names (as detailed in Chapter 8). The converted entity name is included as an extra data element with the tag EntityNameNormalized (along with the EntityNameOriginal).
- It was not felt necessary to create tags for the abstract class of ModelElements, nor for the abstract Relation class (super-type of domain relationships and inheritance relationship).
- For practical reasons, a few attributes of the meta-model were not included in the database. The relevant information is, however, contained in this chapter and/or the appendices. The following three elements are not included in the database:
    - MetaModel
    - SampleDiagram (see problem with string equivalent for pictures)
    - CapturingNotes
- Where copies of the model database are to be made available publicly, i.e. not for personal research purposes only, the definitions should be deleted for copyright purposes.

In the end the following documents were created.

6.4.1 EnterpriseModels.XML

This file contains all data from the enterprise model database, marked up with the appropriate XML tags. The following is a view of the first couple of lines of code with [...] indicating truncated lines.

```
<?xml version="1.0" ?>
<!DOCTYPE ModelDatabase SYSTEM "EnterpriseModels.dtd">
<ModelDatabase>
   <Model>
      <ModelID>AIAI</ModelID>
      <ModelKey>AI</ModelKey>
      <ModelName>The Enterprise Ontology</ModelName>
      <ModelLogo></ModelLogo>
      <ModelDescription>The Enterprise Ontology is [...]
      <ModelAuthor>Artificial Intelligence Applications [...]
      <ModelDateLastChange>1998</ModelDateLastChange>
      <ModelPrimarySourceType>Electronic: KIF format [...]
      <ModelPrimarySourceReference>http://ontolingua.stanford.edu[...]
      <ModelSecondarySources>[USCH98] Mike Uschold, Martin King, [...]
      <ModelReferenceDiscipline>Ontology</ModelReferenceDiscipline>
      <ModelModellingNotation>KIF (text)</ModelModellingNotation>
      <Entity>
         <EntityCode>AIE001</EntityCode>
         <EntityNameOriginal>Activity</EntityNameOriginal>
         <EntityNameNormalized>activity</EntityNameNormalized>
         <EntityDescription>Something done over a particular Time[...]
      </Entity>
      <Entity>
         <EntityCode>AIE002</EntityCode>
etc.
```

The entire database has a size of 8,579,148 bytes and can be viewed with normal text editors or word processors, XML-enabled browsers (the latest versions of most standard browsers will work) or dedicated XML editors. Times given below are purely for illustrative purposes, for a configuration of an IBM-compatible personal computer with a 900MHz Pentium III and 128MB RAM, operating under MS-Windows XP and no other application programs open.

- Microsoft WordPad v 5.1 and Microsoft Word 2002 were used to manipulate and edit the database in raw format. A single search and replace of a short string would typically take between 1 and 5 minutes on the hardware described above. However, the well-formedness of an XML file will only be apparent when loaded in an XML-enabled browser or dedicated XML editor.
- Due to its relatively large size, the database takes a very long time to manipulate on a PC using a web browser. Using Internet Explorer 6.20, the file takes more than 30 minutes to load and display, and manipulation is virtually impossible due to the frequent disk swapping.
- A simple, freeware XML editor, XMLViewer, is made available by Mindfusion. This manages to load the database in well under 40 seconds, consuming about 54 Megabytes of RAM. File manipulation is very quick because it does not construct a screen image of the entire file, as the word processor or browser software appear to do.

Screen images of the XML database as seen in the above software are also contained in Appendix H. A subset containing only 3 of the smaller models is also available for researchers wishing to experiment with the XML database. This improves response times dramatically. Note that XML files are notoriously bloated, partly due to their human-readable tags. Very high compression ratios can be achieved for downloading or transfer purposes. A zipped version of the 8.5 Megabyte database (including the tiny DTD and XSD files) is only 580,522 bytes i.e. a compression ratio of 94%

6.4.2 EnterpriseModels.DTD

Although the syntactical correctness of the EnterpriseModels.XML file can be established without DTD, the DTD ensures that the correct conceptual structure is adhered to. For example, the DTD specifies that each model contains only one ModelName but can have several Entities. If, say, the database were to include a model inside (as part of) an entity, the XML parser would abort because it would contradict the structure as specified in the DTD.

Although the full DTD is listed in Appendix H, the following code extracts illustrate some typical code segments.

```
<!ELEMENT ModelDatabase (Model*) >
```

The ModelDatabase consists of several Models

```
<!ELEMENT Model (ModelID, ModelKey, ModelName, ModelLogo, ModelDescription,
ModelAuthor, ModelDateLastChange, ModelPrimarySourceType,
ModelPrimarySourceReference, ModelSecondarySources,
ModelReferenceDiscipline, ModelModellingNotation, ( Entity | Relationship |
IsaRel | Group )* ) >
```

A `Model` consists of a `ModelID, ModelKey, ... ModelModellingNotation`, and can include any combination (number) of `Entities, Relationships, IsaRels`, and/or `Groups`.

```
<!ELEMENT ModelID  (#PCDATA)>
```

A `ModelID` is Parsed Character Data.

```
<!ELEMENT Entity  (EntityCode, EntityNameOriginal, EntityNameNormalized,
EntityDescription)>
```

An `Entity` consists of an `EntityCode, EntityNameOriginal, EntityNameNormalized` and `EntityDescription.`

```
<!ELEMENT IsaRel  (IsaRelCode, IsaRelSubEntity, IsaRelSuperEntity)>
<!ELEMENT IsaRelCode  (#PCDATA)>
<!ELEMENT IsaRelSubEntity  (#PCDATA)>
<!ELEMENT IsaRelSuperEntity  (#PCDATA)>
```

The section above describes a complete "IsaRel" record.

6.4.3 EnterpriseModels.XSD

It is a fairly trivial exercise to convert the DTD to a well-formed XML schema definition (XSD). Below is a minimal schema for the ModelDatabase. The power of XML schema is that it allows for inheritance (e.g. the IsaRelCode, the EntityCode, the RelationshipCode and the GroupCode would all be inherited from the ModelElementCode) and allows for a better definition of the data types. For instance, the schema could be refined by creating user-defined types for e.g. EntityCodes of the format {2-letter ModelKey}+{the letter "E"}+{a positive integer}. This would enforce a validation of the data against the data type (here acting as a "mask") when loading (and editing) the dataset.

The following is sample code extracted from the XSD, including the first 14 lines as well as the specification for the IsaRel record (complex data type).

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="ModelDatabase" type="ModelDatabaseData"/>

    <xsd:complexType name="ModelDatabaseData">
       <xsd:sequence>
          <xsd:element name="Model" type="ModelData"
                              minOccurs="1" maxOccurs="unbounded"/>
       </xsd:sequence>
    </xsd:complexType>
```

```
<xsd:complexType name="ModelData">
     <xsd:element name="ModelID" type="xsd:string"/>
     <xsd:element name="ModelKey" type="xsd:string"/>
     <xsd:element name="ModelName" type="xsd:string"/>
     <xsd:element name="ModelDateLastChange" type="xsd:gYear"/>
                    [...]
<xsd:complexType name="IsaRelData">
     <xsd:element name="IsaRelCode" type="xsd:string"/>
     <xsd:element name="IsaRelSubEntity" type="xsd:string"/>
     <xsd:element name="IsaRelSuperEntity" type="xsd:string"/>
</xsd:complexType>
```

The full listing of the XSD can be found in Appendix H.

Chapter 7: Syntactic Analysis

Having completed the survey and constructed the database of enterprise models, it is now possible to start with the model analysis by using the framework that was developed in chapter 5. This chapter deals with the first aspect of the framework: syntactic analysis. The next two chapters will deal with the other two aspects of the framework, namely semantic and pragmatic analysis. Syntactic model analysis is concerned with the purely structural aspects of the model, regardless of the underlying meaning of the model and its elements. Most of the analysis presented in this chapter is derived from the disciplines of software engineering, computer science, graph theory and network analysis. Wherever applicable, the appropriate terminology from the reference discipline may be used e.g. "node" or "vertex" for "entity" and "connector", "line", "arc" or "edge" for "relationship".

Full references for the sources of various metrics as well as for their formulae can be found in Appendix B. This chapter starts with the more standard syntactic metrics, such as size and inheritance structure, before moving on to less conventional approaches.

7.1 Model Size

Perhaps the first syntactic measure that is considered for almost any construct, whether conceptual or physical, is its size. Generally, the more complex the entity to be measured, the more different ways exist in which to measure its size.

There is a close association between size and complexity:

> "Size metrics are usually used in conjunction with complexity metrics and the distinction between them is sometimes unclear. [...] Class size metrics, For example, reflect the effort required to build, understand and maintain a class. The same reasoning can be applied to complexity. [...] Size metrics also play an important part in the normalization of composite metrics." [BRIT94, p. 6]

Many different size metrics have been proposed for models. Table 7-1 lists some candidates from the literature [SHEP95; HEND96]. All of these metrics are defined and fully referenced in appendix B and rely on the definitions (e.g. the types of relationships and the grouper concept) as supplied in section 4.4 which discusses the meta-model.

From Table 7-1, it is clear that even a relatively straightforward metric such as "number of entities" becomes problematic; because some models have "orphan" entities which do not participate in any relationships (refer also to the discussion on model correctness below). Quite a few of the models under consideration have a very large number of entities participating in a hierarchical tree structure (a classification structure) but far fewer are connected by means of domain (i.e. not IS-A) relationships. Examples of the latter are AIAI, BelgAcc, Miller and Nippon. It must be noted that the models in the database have slightly different definitions or interpretations of structural relations. Although inheritance and generalisation are not fully overlapping concepts, for purposes of this research they have been grouped together and are denoted as "IS-A" relationships in the discussion which follows.

Table 7-1 lists the statistics for the other model element counts, including the number of relationships, groupers, diagrams and entity attributes. The following are the three main syntactic indicators of model size:

- The *number of model entities* is perhaps the most intuitively acceptable count from a structural point of view. It is useful since it gives an indication of the minimum number of classes that needs to be constructed when implementing the model in an object-oriented information system. It is

also a proxy for the number of tables that may be needed when implementing the model in a relational database environment. The number of vertices is also one of the more common size indicators in network or graph theory, and is equivalent to an entity count. The drawbacks of this measure are that it completely ignores all the other model elements.

Table 7-1: Model Size.

Model	Model ID	Nr of Entities (nodes/vertices)	Nr of Connected Entities excl. IS-A	Nr of Connected Entities incl. IS-A	Nr of (non-IS-A) Relationships = Absolute Connectivity	Nr of IS-A Relationships	Nr of Groupers	Nr of Diagrams	Nr of Attributes	CASE Size or Concept Count	Graph Size	Adjusted CASE Size
AIAI	AI	94	44	91	73	98	5			270	265	510
AKMA	AK	82	56	82	61	33	6	6	383	565	176	769
ARRI	AR	128	119	128	197	70	35	7		430	395	790
Baan	BA	328	232	310	608	142	8			1086	1078	1927
BelgAcc	BE	470	178	470	213	462	13			1158	1145	1158
BOMA	BO	183	128	183	192	115	38	38	24	552	490	770
CYC	CY	777	639	724	1149	654	43			2623	2580	4537
Fowler	FO	120	95	120	131	69	55	55		375	320	579
Hay	HA	291	235	291	725	185	91	91		1292	1201	3465
Inmon	IN	427	225	426	241	220	73	45	1468	2429	888	2670
Miller	MI	48	26	48	81	44				173	173	276
NHS	NH	269	173	269	220	236	1	1	25	751	725	1460
Nippon	NI	147	58	147	58	146	16	1		367	351	483
Ottawa-Big	OB	459	459	459	634					1093	1093	1544
Ott.-Dense	OD	248	248	248	455					703	703	945
Purdue	PU	106	89	89	224		13	12		343	330	866
Random	RA	248	238	247	455	321				1024	1024	1024
SAP	SA	396	328	396	612	169	41	24		1218	1177	1917
Semi-Rand	SR	248	238	247	455	321				1024	1024	1024
SanFran	SF	109	99	109	172	11	40	40		332	292	532
Silverston	SI	267	209	267	322	82	62	55	536	1269	671	2235
TOVE	TO	564	539	550	1216	72	85			1937	1852	2042
USB	US	144	119	144	239	129	19			531	512	770
Total		6153	4774	6045	8733	3579	644		2436	21545	18465	32293
Average		268	208	262	379	179	35		487	937	803	1404

- Slightly more sophisticated is the *graph count* which measures the number of network elements i.e. both the nodes (vertices) and connections (edges). This is easily counted by adding the total number of relationships to the number of entities.

- The *CASE size* (or *concept count*) is a more sophisticated measure which looks at the number of distinct model elements which would be kept in a CASE tool repository. This includes the entities, relationships, groupers, diagrams and entity attributes. The only problem with this measure is that it does not distinguish models that have a similar concept count but contain different degrees of detail for each of the concepts. For example, whether a model just specifies a relationship between two entities or whether it also gives entity role names, cardinalities and a full description of the relationship does not impact the CASE count.

Because the CASE size does not take the amount of detail into account, a further sophistication of the CASE size is suggested which includes the count of the optional attributes of the various model elements as defined in the meta model of this research. This more accurately reflects the number of

non-empty elements in the model database and corresponds quite closely to the number of non-null elements in the XML database.

Table 7-2 lists the ***model size rankings***, using the entity count, concept count and adjusted CASE size measures. For interest's sake, the approximate number of "standard days" (8 full hours) taken to capture the model into the database has also been indicated.

Table 7-2: Models Ranking According to Various Size Metrics.

Ranking Measure / Model	Entity Size (rank)	CASE Size (rank)	Adjusted CASE Size (Rank)	Average Rank	Capture Time (8-hour days)	Available Digitally?
CYC	1	1	1	1.0	1 ½	Yes
TOVE	2	3	5	3.3	2	Yes
Inmon	5	2	3	3.3	2	No
Hay	8	4	2	4.7	3 ½	No
Silverston	10	5	4	6.3	3	Yes
SAP	6	6	7	6.3	3 ½	Yes
Ott.-Big	4	8	8	6.7	½	Yes
BelgAcc	3	7	10	6.7	1 ½	No
Baan	7	9	6	7.3	3	Yes
NHS	9	12	9	10.0	2	Yes
Semi-Rand	11	10	11	10.7	½	Yes
Random	11	10	11	10.7	½	No
Ott.-Dense	11	13	13	12.3	½	Yes
BOMA	14	15	16	15.0	2 ½	Yes
USB	16	16	16	16.0	2	Yes
ARRI	17	17	15	16.3	1 ½	No
Purdue	20	20	14	18.0	2	No
AKMA	22	14	18	18.0	2 ½	No
Fowler	18	18	19	18.3	2	No
Nippon	15	19	22	18.7	½	No
SanFran	19	21	20	20.0	2 ½	Yes
AIAI	21	22	21	21.3	1 ½	Yes
Miller	23	23	23	23.0	2	No

The CYC model is clearly the largest model, although it must be recognized that, despite the fact that the model used for this analysis is a subset of the entire CYC knowledge base, not all concepts in CYC are directly related to the enterprize. Following closely behind are TOVE and Inmon. TOVE is a fairly intricate model in formal notation, whereas Inmon's model is more of a "data warehouse repository dump" and only makes it this high due to its large number of entity attributes. All of these large models have concept counts of about 2000 or more.

Depending on the exact metric used, these large models are followed by two proper "data models", consisting of fewer entities but specified with much greater detail: Hay and Silverston. These are followed by a cluster of models that are fairly similar in size: SAP, Baan, BelgAcc, NHS and both Ottawa and random models. Their concept counts all are just above 1000.

The next group consists of the more compact models of BOMA, USB, ARRI, Purdue, Fowler and Nippon, with concept counts from roughly 350 upwards to twice that. Although there is no sharp cut-off line, these are followed by smaller models such as Nippon, SanFran, AIAI and Miller.

The above table shows that there is a fair degree of correspondence between the three size measures (and indeed a very high correlation coefficient), though there are a number of discrepancies. Both the Silverston and the Hay models move up quite a few places if a more sophisticated size metric is used. Their raw entity count is misleading because, although they have fewer entities than some of the

larger models, their model specification is very detailed. BelgAcc, Ottawa and Nippon are much "shallower" models, basically consisting of lists of entities and relationships without any description. Hence they contain a relatively large number of model elements but do not specify them to any degree of detail. This supports the value of a more sophisticated size metric.

The capturing time for each of the models has also been included, rounded to the nearest half-day (4 hours). There appears to be little correlation between any of the size measures and the time that was required to capture the model into the model database. This can be confirmed by calculating the Spearman's rank correlation coefficient: $r' = 0.134$ which, though positive, is far from statistically significant. Since some models are available in electronic format and their capture time really measures the format conversion time, the rank correlation was computed for only those 10 models which had to be captured from scratch, but for that subset the rank correlation was even smaller with $r' = 0.115$, still positive but not statistically significant.

7.2 Correctness and Consistency

One of the most important model quality attributes is model correctness. As shown in Chapter 5 and Appendix M it features prominently amongst the evaluation criteria from most authors. Chiorean *et al.* describe *correctness* as follows:

> "By model correctness, we understand the correctness of the model against the modeling language. Model correctness is certainly a very important aspect unfortunately ignored by many specialists in the modeling domain. How else could we possibly explain a series of errors found in different UML models?" [CHIO02, p. 71]

Note that the above description is a syntactic interpretation and generally used in computer science. As discussed in Chapter 5, many business analysts and other authors extend the correctness criterion to include also a semantic component, namely how correctly the model portrays the domain. For clarity reasons, and as per framework overview in Chapter 5, the syntactic definition will be adopted in this research, whereas the term "model validity" will be used to measure the semantic correspondence of a model with its domain. A good example of applying graph-theoretical principles to model analysis is given in Nilakanta [NILA90] who uses graph-theory to check DFDs for correctness.

In this research, two aspects of correctness will be investigated:

- The occurrence of any inconsistencies or errors in the model description as discovered during the model capturing process. This may not be a complete listing because some errors may have escaped detection.
- The existence of and adherence to internal standards evident in the model. This can also referred to as *consistency*.

Measuring model correctness requires that the modeller uses some formal notation or structure, e.g. Miller describes his model in (English) prose and, apart from grammatical and spelling mistakes. This makes it virtually impossible to pinpoint correctness errors.

Most models that use some type of formal notation have been produced using modelling tools which should eliminate most if not all of the syntactic errors. For instance, the Rose Validator tool flags the following warnings for a UML diagram modelled in Rational Rose: no association for message; association not correctly navigable; no class for object instance; no operation for message; broken associations; broken generalizations; broken "has" relationships; broken dependency relationships;

broken instantiates relationships; broken "realize" relationships; broken diagram icons and unused operations. Hence it should be expected that correctness problems in public models are rare. Consequently, inconsistencies and errors will be judged fairly harshly.

Because the various enterprise models use very different modelling notations and differ substantially in formality, no standard and universal list of correctness tests (such as the tests by the Rose Validator tool) can be used and hence no *systematic* way of ascertaining correctness problems is therefore suggested in this research. Where the framework is to be applied in a more structured or standardized context (i.e. different models but using the same modelling notation or tool), it would be a fairly trivial exercise to adopt a checklist of possible correctness issues as suggested by the definition of, or further academic literature on, the modelling language. However, the "discovery of inconsistencies" was not an entirely haphazard or ad-hoc process. The model capturing process necessitated the conversion of each model to a single standard i.e. the lowest common denominator meta-model described in 4.4. This guarantees the identification of any inconsistencies because these result in the necessity for changing the capturing process. This can be illustrated by the following examples. For the automated conversion of an electronic model, all deviations from the standard conversion method have to be coded explicitly e.g. for TOVE, separate search strings have to be defined to map both "defrelation" and "define-relationship" to the same meta-model element. For the manual conversion of, say SanFran or SAP, the data-capturer has to identify the degree of overlap between each diagram in order to capture unique model elements only, and will thus systematically detect inconsistencies between diagrams.

For purposes of this research, the following types of errors or problems were checked.

- Correctness: Do orphan entities (i.e. entities not participating in any relationship) exist? Are there circular inheritance or IS-A relationships? Are links between different pieces of information (e.g. an entity and its definition elsewhere) correct? Are attributes moved up to the highest possible node in the inheritance tree? Are text labels (i.e. model element names) unique within the model and consistent across diagrams?
- Consistency: Are entities and relationships always defined in the same way? Are attributes always listed for each entity and/or relationship? Does information from one diagram conflict with the information of the same modelling element on another diagram?
- Standards: Do names of entities and relationships always follow the same naming convention e.g. concatenation, capitalization, use of plurals? Is the same modelling notation used throughout?

Not all "problems" rate equally from both a practical and theoretical aspect. The insertion of a "blank space" in a concatenated entity name or the omitting of a relationship cardinality is unlikely to produce problems for a human reading a diagram but may well be problematic when used in an automated tool. On the other hand, most proper modelling tools will enforce a degree of correctness e.g. the correspondence of model element names and the relationship attributes across different diagrams. The differing *degrees of correctness* ultimately depend on intended use of the model. If a model is to be used as the basis for an actual database design, entity duplication or incorrect relationship cardinality will have a significant impact. In modelling tools, correctness tests will often distinguish between critical problems and warnings. Because the intended use of the models is not know a priori in this research, a more subjective distinction between "major", "medium-level", "minor" and "no problem" has been used.

A final methodological remark concerns what appear to be semantic errors but are in fact syntactic errors. A correctness test above implies that each model element must have a unique identifying name

– which is usually implemented as a text label in modelling tools. This is a syntactic requirement. If, as happens in the SanFran model, an entity is given a certain label (say "Party" on a UML class diagram and this same entity is given a (somewhat) different label (say "Business Partner") in the corresponding UML sequence or instance diagram, then that is a syntactic (correctness) problem. It must be acknowledged, however, that this may require some semantic interpretation on behalf of the data capturer who will identify the context and infer that the same class is referred to in both diagrams. However, this is usually derived from a purely syntactic perspective because of the unique (binary) relationships in which they partake: if a (unique) relationship with the name "ExternalCommunicationsResponsibity" links "Employee" with "Party" on diagram A and links "Employee" with "Business Partner" on diagram B, then no semantic interpretation is necessary to derive the correctness problem and the semantic meaning of the text label "Party" might as well have been named "AD424". A similar problem was found in the USB model where *different* entities were given the same name. This was found because the definition (either by means of formulas or actual values) of the entities was different but the entities themselves were given the same name. For example, the entity with the label "Accounts Receivable" in the balance sheet were defined as an asset (calculated as a percentage of sales) whereas the entity with an identical label "Accounts Receivable" in the cash (in)flow statements was given a very different definition namely the difference between two subsequent Asset "Accounts Receivables" (month/year-end) balances. Again, the fact that different definitions were used (and the entities participate in different relationships) is a syntactic issue – no semantic interpretation of the actual entity name is necessary. Note that the model also contained multiple instances of the asset entity "Accounts Receivables" which were easily identified and distinguished from the cash flow version by syntactic means.

Table 7-3 summarizes the correctness errors and inconsistencies which were discovered in the various models during the capturing process; more detailed descriptions and examples are provided in Appendix A. As noted above, some inconsistencies and errors could have escaped my attention.

Table 7-3: Correctness Problems and Inconsistencies.

Model	Inconsistencies and errors found	Incorrect Severity	Consistent Severity	Overall level of standards	Score
SAP		none	none	high	8
AIAI	1 orphan entity: "Segmentation-Variable"	minor	none	high	7
BOMA	Minor inconsistencies e.g. chapter on "process objects" lists attributes and methods and names relationships but other chapters do not list attributes and name very few relationships.	none	minor	high	7
Hay	Two minor discrepancies found: • 10.7 "Conditions/Setting" versus 10.5 "Conditions" and 10.9 "Material" should be "Material type" as used in all previous diagrams.	minor	none	high	7
BelgAcc		none	none	medium	7
Baan	Slight naming inconsistencies (e.g. "line of business" singular whereas all other tables & in-text reference are plurals). Fig 14.2 incorrectly reproduced: it is not the financial customer group but match invoice with order. A number of configuration parameters are set but not referred to in subsequent tables resulting in orphan entities.	minor	minor	high	6
CYC	Many orphan entities e.g. UnemployedPerson, TimesPerMinute, TimesPerWeek, StockType, Railway etc. usually caused by omission of the #$ prefix.	minor	minor	high	6

Fowler	A few contradictions (e.g. cardinalities in fig 3.10: Observation 0..* <-> 0..1 Protocol versus fig 4.7: Observation 0..* <-> 1..1 Protocol). Role names for very few relationships.	minor	minor	high	6
AKMA	Some undefined classes e.g. "TRANSFER" and "POST CODE CLASSIFICATION GROUP COMPOSITION". Some incorrect hyperlinks (incorrect file names). Some classes have no attributes. Inconsistency in relationship names (some have an initial capital, most do not).	medium	minor	high	5
ARRI	Two undefined flows: "resource" and "spare". Confusing use of single and plural entity names: "requirement" and "requirements" are defined as different entities.	minor	minor	medium	5
NHS	Two relationships are defined in the "from" entity but not in the "to" entity. A large number of subtypes are listed under the supertype without a corresponding defining entry for the subtype. 17 relationships are defined with contradictory cardinality information at either end (e.g. "expected restart" 0..1 versus 1..1 at "Timepoint" entity).	major	none	high	5
Nippon	Some processes referred to, are not defined (8.1.4 and 7.6). Repeating headers in process tables (e.g. "Marketing and Sales").	medium	none	medium	5
TOVE	Inconsistent way of defining concepts: some sub-ontologies use the "defrelation", other use "define-relationship". Inconsistent naming of concepts and relationships esp. actions. Quite a few orphan entities.	minor	medium	high	5
USB	Same name used for different concepts e.g. the label "Accounts Receivable" is used both for the variable measuring the asset (month-end balance) and the cash (in)flow from that asset over a month or year. ("Accounts Payable" and "Dividends Payable" are similarly used with different meanings).	minor	minor	medium	5
Random	The random relationship generator resulted in a circular inheritance relationship as well as an "orphan" entity aluminium.	major	none	low	3
Purdue	Two non-directional data flows. External entities on lower-level diagrams that do not exist on the higher level diagrams (e.g. assigned work crews). Many unlabelled data flows. Inconsistent labelling of inter-diagram flows (Analysis data = lab analysis?), incorrect labels (3.1 to 3.2.2 should be 8.1 to 3.2.2), missing corresponding legs of inter-diagram flows (5 instances).	major	major	medium	1
SanFran	Much inconsistent entity naming between diagrams (e.g. "PartyRole" = "Business Partner Role"; "QuantityUnit" versus "Quantity Unit". Quite a few spelling errors (e.g. in "CurrencyGailLossAccounts.gif" (?) : "UrealizeLoss"). Inconsistencies in naming (e.g. concatenation: "PaymentMethod" but "Payment Plan" and "Payment Detail").	medium	major	low	1

Inmon	Inconsistent use of delimiters ("date (actual)" versus "discounts – quantity" versus "supplier/order"). Ad-hoc attributes. Address specification differs for different entities. Inconsistent use of options e.g. "receipt / no receipt" vs. "follow up (yes/no)" vs. "shared?". Many spelling mistakes e.g. "quaterly" ; "small buseness". Numerous conflicts between the seven high-level models (e.g. customer in sales: credit rating = commercial customer attribute versus customer in marketing: credit rating = (generic super-class) customer attribute). Incorrect use of inheritance (e.g. "Snapshot date" attribute in commercial & individual customer should move up to the parent class "customer"). Repeated fields/attributes (e.g. "change date" appears twice in "change in accounting year"). Inconsistent naming of data elements. One orphan entity: "IRS contact" (forgot to draw connector line?).	major	major	low	0

The models are rated in decreasing order of correctness, with a scoring system for both correctness and consistency problems of 3 for no problems, 2 for minor, 1 for medium and 0 for major problems. In addition, a score of 2 was allocated for models which had fairly stringent standards for naming, diagramming etc., 1 for medium level standard and 0 for a low level of standards. These add up to a combined "correctness score" with a range from 0 to 8 – admittedly somewhat arbitrary – where the higher the score, the better the correctness. The natural language models Miller, Ottawa-Big, Ottawa-Dense and Semi-Random were not included since natural language is not formal enough to rate model correctness, although the Random model was included to illustrate the correctness problems associated with randomly generated relationships.

- From the above analysis, it appears that the only model that could not be faulted on any of the criteria was the SAP model, although models such as AIAI, BOMA, Hay, BelgAcc, Baan, CYC and Fowler also rate high on overall correctness, consistency and adherence to standards.
- Some correctness problems are experienced by AKMA, ARRI, NHS, Nippon and TOVE, and USB.
- The Purdue and SanFran models have major problems and the Inmon model has the lowest possible correctness score.

Virtually all of these correctness problems can be remedied or avoided by using a proper modelling tool, so the rating was applied fairly harshly. It is argued here that publicly available models should really not have any correctness problem.

7.3 Hierarchical Structure and Grouping

Models have a distinct structure, which can be expressed by three different constructs: diagrams, groupers and inheritance tree.

7.3.1 Groups and Diagrams

Most models group similar entities in logical sub-units or chunks, e.g. by functional area of the organization. This is indicated by means of grouper constructs which are usually focused on the collection of like *entities* e.g. all financial entities. Relationships between entities are often *not* considered to belong to specific groupers, since many link concepts belong to different groups.

This is slightly different to diagrams. Where a model consists of visual diagrams, each diagram will typically also group related entities. However, most models prefer to re-draw a given entity on a number of different diagrams than to draw a relationship that crosses diagrams. Thus diagrams will generally emphasize (and group together) the *relationships* between entities on the same diagram.

The distinction between grouper and diagram is not always clear-cut. For a number of models, the diagram is the only logical grouper. Other models do not use groupers but rely on the inheritance structure to group entities. Sometimes, the concepts of grouper and inheritance get conflated into a tree-like structure such as the account structure of the BelgAcc model. In other cases, such as CYC, there is a more complicated hierarchical structure because of the frequent use of the "type" construct, which has other classes as *instances* instead of subclasses. This implies two types of structural relationships: the usual "Is-A(-Type-Or-Kind-Of)" as well as a "Is-An-Instance-Of" relationship (which is different from the instantiation method which creates object instances!)

It is possible to calculate a large number of metrics relating to the number and relative size or density of groups and diagrams. Not all of them are equally valid: some models have only a very small number of groups; others are purely textual and do not use diagrams at all. As pointed out above, it usually makes more sense to calculate the average number of entities per grouper construct and the average number of relationships per diagram, than the other way round. Note that the two random and the two Ottawa models are not listed since they use neither grouper constructs nor diagrams.

Table 7-4: Grouper and Diagram Metrics.

Model	Modeel ID	Nr of Entities	Nr of Relationships	Nr of Groupers	Nr of 1st Level Groupers	Nr of Grouping Levels	Avg nr of entities/group	Nr of Diagrams	Avg nr of entities/diagram	Avg nr of relation-ships/diagram
AIAI	**AI**	94	73	5	5	2	18.8			
AKMA	**AK**	82	61	6	6	2	13.7	6	13.7	10.2
ARRI	**AR**	128	197	35	6	2	3.7	7	18.3	28.1
Baan	**BA**	328	608	8	8	3	41.0			
BelgAcc	**BE**	470	213	13	3	4	36.2			
BOMA	**BO**	183	192	38	4	3	4.8	38	4.8	5.1
CYC	**CY**	777	1149	43	43	2	18.1			
Fowler	**FO**	120	131	55	6	3	2.2	55	2.2	2.4
Hay	**HA**	291	725	91	8	3	3.2	91	3.2	8.0
Inmon	**IN**	427	241	73	7	3	5.8	45	9.5	5.4
NHS	**NH**	269	220					1	220	220.0
Nippon	**NI**	147	58	16	4	5	9.2			
Purdue	**PU**	106	224	13	12	3	8.2	12	8.8	18.7
SAP	**SA**	396	612	41	17	3	9.7	24	16.5	25.5
SanFran	**SF**	109	172	40	40	2	2.7	40	2.7	4.3
Silverston	**SI**	267	322	62	7	3	4.3	55	4.9	5.9
TOVE	**TO**	564	1216	85	11	4	6.6			
USB	**US**	144	239	19	19	4	7.6			
Average		268	380	72		3.0	11.5		37.6	32.6

Note that in Table 7-4, averages refer to the average number of *unique* entities/relationships per grouper/diagram. A rule of thumb is that short term memory can hold only 7 distinct concepts (plus or minus 2), hence the ideal chunk size should range between 5 and 9 [WITT94]. However, many model groupers or diagrams are used for reference or documentation purposes rather than for introducing new conceptual structures, so it is not clear that this cognitive limit is valid.

The interpretation of the above metrics should be done with great care, taking into account the specific context for each model. The following are some cautionary illustrations of the relative uselessness of the above measures:

- The Baan model appears to have a much higher "entity/group" ratio than e.g. SAP, but that is mainly due to the fact that the Baan model was captured (re-engineered) from a book organized

around the different modules of the Baan software package, whereas the SAP model was derived from the original conceptual model, organized around summary diagrams for functional (sub)areas. The actual SAP R/3 implementation also consists of a relatively small number of modules, comparable to Baan, and would have the same entity/group ratio if the model was extracted in a way similar to Baan.

- NHS has a ridiculously high ratio of 269 (unconnected) or 220 (connected) entities/diagram. This is because the only electronically available documentation for NHS is the "wall-chart" containing the entire model on one sheet. Because of its different purpose, this should obviously not be compared to the other models.
- The second-highest numbers of entities per diagram is ARRI, 18.3, but there are actually never more than 6 or 7 *ICOM* entities per diagram. The other entities are "artificial" input, output or control entities which occur on the diagrams as multi-node connectors. Depending on the meta-model used for data-capture, these could be considered to be either relationships or entities. SAP has an average number of (new) entities per diagram of 16.5, the second highest number excluding the exceptional case of NHS. However, in practice, the SAP diagrams are much more cluttered than ARRI's because a large number of entities are repeated across different diagrams. Admittedly, SAP tries to reduce the cognitive overload by applying gray-shading to the newly introduced entities, but this does not reduce the fact that its diagrams are by far the densest of all models.
- At the other end of the spectrum are Fowler and SanFran whose diagrams are the sparsest (in terms of both entities and relationship per diagram). Although their ratios might be considered much too low (as compared with the "magical seven plus or minus two"), it should be considered that both are concerned with high-level patterns and these take considerable mental effort to assimilate.

Overall, the grouper and diagram metrics are not very useful for purely analytical comparative analysis purposes, unless substantial additional explanations are added.

7.3.2 Metrics for the Inheritance Structure

The literature on OO metrics is replete with inheritance metrics. Although most of them require constructs found only in the design phase (e.g. methods), there are quite a few that are also appropriate at the analysis level.

OO and conceptual modelling in general support *multiple inheritance*, and a number of OO languages such as C++ and Smalltalk implement this. However, Lorenz & Kidd, as well as many other practitioners [PRES97c], do not recommend the use of multiple inheritance in modelling businesses and consider it "an anomaly" to be used only in exceptional cases and as a very conscious decision.

Fenton [FENT96] suggested the following rather simple morphology metrics, which summarize the shape of the model inheritance graph (tree).

Graph Size = Entities Count + Relationships Count

Connectivity Density = Arc-to-Node Ratio = Relationships Count / Entities Count

Depth = Longest path from root (top) node to a leaf node

Width = Maximum number of nodes at any one level of the architecture

The *class hierarchy nesting level* is a count of the maximum number of levels in the inheritance hierarchy. As Lorenz & Kidd point out, this is likely to be higher when OO frameworks or libraries are used. They suggest that the maximum threshold should be *six* (from the top of the class hierarchy or the bottom of the framework) and claim that large nesting numbers indicate a design problem. IBM's BSDM methodology recommended that the depth of an entity model should be no more than 4 layers, i.e. 4 steps through parent links [CHEN98].

Binder's *Number of Root Classes* [PRES97c], NOR, counts the number of distinct class hierarchies. Most of the above inheritance metrics should be calculated for each root class hierarchy but can also be calculated for the model as a whole (including or excluding classes that fall outside any inheritance hierarchy).

The *average depth of the inheritance tree* assumes that all classes belong to a single or number of parallel inheritance trees. It is calculated by determining the depth of each class in its hierarchy (misleadingly called DIT or Depth in Inheritance Tree) - referred to as the nesting level by Lorenz & Kidd, or "class-to-root" depth by Tegarden and Sheetz.

$$\textit{Average Inheritance Depth} = \text{AID} = \sum \text{DIP}_i \;/\; \text{total number of classes}$$

A small but methodologically sound empirical investigation into OO metrics showed that the Number of Children (NOC) and Depth of Inheritance Tree (DIT) metrics are highly significant predictors for errors occurring in the development of systems. However, the typical NOC for the systems was less than 3 and DIT less than 4. [BASI96]

In order to evaluate the extent to which the designer re-used classes with hierarchies, Yap & Henderson-Sellers suggested the following two metrics: the reuse and the specialization ratios. [HEND96]

$$\textit{Reuse Ratio} = \text{U} = \text{number of superclasses} \;/\; \text{total number of classes}$$

The reuse ratio indicates the extent to which the designers (or prospective implementers) of the class library have been (or will be) able to inherit from their own classes to create new classes. U is always less than one, but gets closer to one when there is a large number of classes and a fairly linear hierarchy, or when multiple inheritance is used a lot. Conversely, a value near 0 indicates a shallow depth.

The second of their ratios to measure re-use is:

$$\textit{Specialization Ratio} = \text{S} = \text{number of subclasses} \;/\; \text{number of superclasses}$$

The specialization ratio measures the extent to which a superclass has captured the abstraction. A large S indicates a high degree of reuse since there are lots of subclasses. With multiple inheritance, the value can be less than 1 although generally values of S (and U) close to 1 suggest poor design.

Table 7-5 lists the metrics for the various models, as defined above.

Table 7-5: Inheritance Metrics.

Model ID	Nr of IS-A Relationships	Multiple Inheritance?	Nr Unique Entities in Inheritance Graph	Inheritance Graph Size	Inheritance Connectivity Density	Inheritance Depth / Hierarchy Nesting Level	Mean Depth of Inheritance Tree	Inheritance Width	Widest Level (1 = Top)	Nr of root classes	Average Number of Children	Standard Deviation NOC	Nr of superclasses	Reuse Ratio	Specialization Ratio
AIAI	98	Y	89	187	1.10	6	3.2	37	3	2	3.4	4.2	29	31%	2.07
AKMA	33	N	39	72	0.85	2	2.3	21	2	7	3.3	2.1	10	12%	2.90
ARRI	70	N	82	152	0.85	3	2.4	42	2	12	3.9	1.9	18	14%	3.56
Baan	142	N	167	309	0.85	4	2.1	122	2	36	3.4	2.0	42	13%	2.98
BelgAcc	462	N	470	932	0.98	4	3.1	253	3	8	4.1	2.3	114	24%	3.12
BOMA	115	N	123	238	0.93	3	2.9	47	3	14	3.3	1.4	35	19%	2.51
CYC	654	Y	526	1180	1.24	10	4.0	181	2	54	2.8	3.0	233	30%	1.26
Fowler	69	N	87	156	0.79	3	2.5	42	2	18	2.5	1.5	28	23%	2.11
Hay	185	Y	192	377	0.96	3	2.6	106	2	27	3.4	2.4	54	19%	2.56
Inmon	220	N	242	462	0.91	3	2.1	194	2	24	7.3	6.4	30	7%	7.07
Miller	44	N	48	92	0.92	5	2.7	21	2	4	3.7	2.4	12	25%	3.00
NHS	236	N	247	483	0.96	5	3.9	69	5	12	5.2	3.2	45	17%	4.49
Nippon	146	N	147	293	0.99	5	4.3	73	4	1	4.1	2.1	36	24%	3.08
Random	321	Y	230	551	1.40	13	4.7	75	2	46	1.7	0.8	189	76%	0.22
SAP	169	Y	151	320	1.12	3	2.2	121	2	25	4.3	4.3	39	10%	2.87
Semi-Rand.	321	Y	230	551	1.40	13	4.7	75	2	46	1.7	0.8	189	76%	0.22
SanFran	11	N	17	28	0.65	2	2.1	11	2	7	1.6	0.9	7	6%	1.43
Silverston	82	N	97	179	0.85	3	2.4	59	2	18	3.2	1.4	26	10%	2.73
TOVE	72	Y	80	152	0.90	4	2.2	54	2	13	3.3	4.2	22	4%	2.64
USB	129	N	144	273	0.90	2	2.0	129	2	15	8.6	12.9	15	10%	8.60
Total	3579		3408	6987											
Average	179		170	349	0.98	4.8	2.9	87	2.4	19	3.7	3.0	58.7	23%	2.97

The following comments are organized by metric.

- CYC, by far the largest model, also has the largest number of inheritance relationships as well as the greatest depth, whereas SanFran makes by far the least use of inheritance.
- According to [PRES97c; HEND96], most practitioners suggest that multiple inheritance should be avoided or used extremely sparingly. Clearly, many enterprise models do not follow this guideline. A possible explanation is that, whilst multiple inheritance leads to significant practical implementation issues, it appears to be a very useful construct in the conceptual analysis of a domain, especially when dealing with a fairly abstract domain.
- It is interesting to note that the two models with the highest inheritance connectivity density are both models from an ontology origin. One of the main foci of ontologies is to create a strong hierarchical structure of the concepts within the domain. The third model of ontology origin, TOVE, also has a relatively high density. It is, however, a great surprise to discover that the two models with the lowest density (<0.80) are Fowler and SanFran, both of whom are from the patterns discipline. Also surprising is the relatively large difference between the SAP and the Baan model. This is probably indicative of the fact that many abstract super-classes do not find their expression in the implementation of a (relational database) system. Remember that SAP was

captured from the conceptual model whereas the Baan model was re-engineered from the ERP implementation.

- The inheritance depth should not be too deep, otherwise model understanding and debugging is impeded. Clearly CYC has got a large problem, although its ten levels must be seen in the context of the many abstract super-classes (bordering on meta-classes) it uses: "thing", "object", "material object" etc. Note that the structure of a natural domain is very different from a randomly generated set of inheritance links: the random model incorporates some random generalization links which result in a 13 level inheritance depth. For the remainder of the inheritance discussion, the random models will be ignored.
- A more representative metric for inheritance depth might be the mean depth of the inheritance tree. Values above a mean depth of three are rare, as is the case for AIAI and CYC, also the two densest trees. The figures for BelgAcc, NHS and Nippon reflect their strong hierarchical structure, which receives less emphasis in other models.
- The inheritance width is a fairly meaningless construct, as is the widest level. The former is highly dependent on the size of the inheritance tree, whereas the latter depends very much on the number of top or root classes.
- The average and standard deviation of the number of children is very high for Inmon and the USB model. This suggests that perhaps an additional hierarchical level might be recommended. At the opposite end of the scale are SanFran and Fowler, models whose narrow width reflects their relatively low inheritance density.
- Although there is no ideal reuse ration, values close to 0 are said to be indicative of *shallow* models. This is indeed the case for Inmon (data warehousing) and SanFran (patterns), although TOVE's position is harder to explain. At the top (>30%), the dense, knowledge-based AIAI and CYC models are found.
- Finally, as pointed out in Appendix B, specialization ratios close to 1 are said to be indicative of poor design. It is surprising to see CYC and, less so, SanFran labelled as such. However, a small vindication is the extremely low values for the random models (the specialization ratio can be less than 1 only because of multiple inheritance). Overall, this ratio does not seem to be very illuminating since there is no ground to assume that USB, Inmon and NHS are better designed than any of the other models.

In conclusion, it can be said that the inheritance ratios give a fair feeling for the shape of the inheritance tree, and the degree to which inheritance is used in a model, but the interpretation of the metrics for comparing models leaves much to be desired. The validity of these metrics at this level of analysis should be questioned, though there may well be a different case for their use at the design level.

7.4 Complexity[4] and Density

The literature on metrics proposes a multitude of systems design (and a smaller number of analysis) metrics. Some of these can be applied at the high-level business model level. For example, [BRIT94] suggests the following complexity metrics.

- ***Method Complexity (MC) metrics***

[4] This section on complexity was presented at the SAICSIT 2002 conference and published as [VANB02].

- **Cyclomatic complexity** - as defined by Tom McCabe (McCabe, 1976)
- **Volume metric** - as defined by Maurice Halstead (Halstead, 1977)
- **Information flow** - as defined by Sallie Henry and Denis Kafura (Henry et al., 1981)

- *Class Complexity (CC) metrics*
  - **Children count** - is the number of directly inheriting sub-classes
  - **Progeny count** - is the number of sub-classes that inherit directly or indirectly
  - **Parents count** - is the number of super-classes from which the class under consideration inherits directly
  - **Ascendancy count** - is the number of super-classes from which the class under consideration directly or indirectly inherits
- *System Complexity (SC) metrics*
  - **Total length of inheritance chain** - is the total number of edges in the inheritance hierarchy graph
  - **Coupling between objects** - is the number of non-inheritance related couples with other classes, where coupling is a measure of two objects acting upon each other. This metric is detrimental to modular design and prevents reuse; it also necessitates more complex testing because of the complex

A more detailed and critical discussion of the more important and relevant complexity metrics can be found in [ROSS96], who investigated the complexity of methods, i.e. method metrics, by looking at the number of concepts in the method meta-model. To this end, they represented a meta-model of a method with entity relationship model and calculated the number of the entities, of the relationships, etc.

Most of the metrics discussed by [ROSS96; BRIT94] and others have already been discussed above. The table below shows some commonly calculated model complexity metrics namely cyclomatic complexity, relative connectivity, fan-out and data bang. Any relative or average measure, such as relative connectivity or average fan-out, can also be said to measure model **density**.

There are many more statistics relating to the use of inheritance and diagrams, but these are not applicable to all models in the test bed since a number of the models do not incorporate inheritance relationships and others have no graphical diagrams. In the table, only entities participating in at least one "domain" (i.e. non-inheritance) relationship were used. The following are brief descriptions of the metrics.

The **cyclomatic complexity** is the classic complexity metric proposed by McGabe [MCGA76] to measure program complexity. It is also useful for measuring model complexity, since it was originally derived from graph theory [SHEP95]. It is calculated as:

$$CC = \text{Number of arcs} - \text{Number of vertices} + \text{Number of disjoint graph partitions}$$

From the list of 48 syntactic complexity measures as given by [EDMO99], it is the most efficient and applicable. Kolewe [KOLE93] proposed a slightly different version under the name of "**Association Complexity**", using a weighting of 2 for the number of disjoint graph partitions. Kolewe's association complexity explicitly excludes inheritance relationships and groupers.

Apart from Cyclomatic Complexity, [EDMO99] also cites the total number of relations as a measure for **connectivity** i.e. the extent of inter-connections between model components as a means to gauge model complexity. Since this is heavily influenced by the model size, one can also calculate the number of relations relative to the number of entities.

$$\text{Absolute Connectivity} = \text{Number of Relationships} \quad \text{and}$$

$$\text{Relative Connectivity} = \text{Number of Relationships} / \text{Number of Entities}$$

Relative connectivity can be calculated including or excluding the inheritance relationships.

Both Kitchenham [SHEP95] and Chidamer & Kemerer [HEND96] proposed **fan-out** as metrics. Kitchenham's original measure referred to subordinate (program) modules but recognized an orthogonal version based on (directed) relationships. In this research, an *entity's fan-out will refer to the number of relationships in which it participates*. In a directed graph, this can be as an ending or originating node. Kitchenham cites both the "average" and "highest" fan-out values as important measures. Chidamer & Kemerer refer to "coupling between object classes" and allowed distinctions based on the type of reference. Henderson also stressed the importance of checking the distribution (e.g. standard deviation) of the fan-out values.

DeMarco suggested two "**bang**" metrics [SHEP95], with **Data Bang** being the one most relevant to data models. It is calculated using the following formula:

$$\text{Data Bang} = \sum \text{COBI}_i = \sum \text{RE}_i \text{ (Nr of relationships for } i^{th} \text{ entity) x } w_i \text{ (} i^{th} \text{ entity's weighting)}$$

COBI stands for "Corrected OBject Increments". The weighting factors are according to a table provided by DeMarco. Although DeMarco suggests that the metric is a "quantitative indicator of net usable function from the user's point of view", Sheppard disagrees and suggests that additional empirical research evidence is required. MacDonell [1994] finds slightly more merit in the measure when comparing it to others, but agrees on the lack of validation.

Table 7-6: Complexity Measures.

Model ID	Nr of Connected Entities /nodes/vertices	Nr of Relationships/Arcs = Absolute Connectivity	Cyclomatic Complexity	Relative Connectivity excl. IS-A relationships	Relative Connectivity incl IS-A relationships	Average Fan-Out	Maximum Fan-Out	StdDev of Fan-out	Data Bang	Avge Data Bang
AIAI	44	73	30	0.78	1.82	3.32	12	3.09	220	4.99
AKMA	56	61	6	0.74	1.15	2.18	9	1.60	160	2.85
ARRI	119	197	79	1.54	2.09	3.31	15	2.85	592	4.97
Baan	232	608	377	1.85	2.29	5.24	43	5.94	2018	8.70
BelgAcc	178	213	36	0.45	1.44	2.39	54	4.43	599	3.37
BOMA	128	192	65	1.05	1.68	3.00	15	2.58	557	4.35
CYC	639	1149	511	1.48	2.32	3.60	105	5.48	3507	5.49
Fowler	95	131	37	1.09	1.67	2.76	12	2.37	372	3.92
Hay	235	725	491	2.49	3.13	6.17	73	9.12	2470	10.51
Inmon	225	241	17	0.56	1.08	2.14	23	2.99	682	3.03
Miller	26	81	56	1.69	2.60	6.23	21	5.12	272	10.47
NHS	173	220	48	0.82	1.70	2.54	35	3.56	622	3.60
Nippon	58	58	1	0.39	1.39	2.00	8	1.49	149	2.56
Ottawa-Big	459	634	176	1.38	1.38	2.76	24	2.99	1830	3.99
Ottawa-Dense	248	455	208	1.83	1.83	3.67	22	3.04	1359	5.48
Purdue	89	224	136	2.11	2.11	5.03	15	2.68	711	7.99
Random	238	455	218	1.83	3.13	3.82	10	1.91	1355	5.69
SAP	328	612	285	1.55	1.97	3.73	73	4.89	1851	5.64
Semi-Random	238	455	218	1.83	3.13	3.82	10	1.91	1355	5.69
SanFran	99	172	74	1.58	1.68	3.47	12	2.84	520	5.25
Silverston	209	322	114	1.21	1.51	3.08	26	3.60	950	4.55
TOVE	539	1216	678	2.16	2.28	4.51	127	7.81	3876	7.19
USB	119	239	121	1.66	2.56	4.02	12	2.80	740	6.22
Total	4774	8733							26766	
Average	208	380	173	1.40	2.00	3.60	33	3.70	1164	5.50

Table 1: Software engineering metrics for enterprise models

A more detailed analysis is possible when the selection of models is restricted to those of similar modelling approaches. For object-oriented models, it is possible to compute a large number of "complexity" and inheritance measures. Similarly, a large number of diagram-related metrics can be computed for models in graphical notation.

Space limits a systematic and in-depth discussion of each metric. The following highlights the deficiencies and inadequacies of the metrics by means of prototypical examples:

- The number of entities and relationships gives a very rough indication of model size (in effect they are a subset of the "CASE count" metric) but gives no indication whatsoever of model complexity or quality. For instance, the Ottawa-Big model appears much larger than the much more complex and well-validated SAP R/3 reference model. Inmon is roughly the same size as Silverston but the latter is of much higher quality, sophistication and density.

- The cyclomatic complexity is rather meaningless since the measure is not normalized for model size. The random model rates the same as SAP. The more complex SanFran, AKMA and Fowler receive unwarranted low ratings.

- Relative connectivity and average fan-out give a fairly good insight of relative model density or connectivity, but are greatly skewed by outlying fan-out values. For example, detailed model inspection shows that the SAP model is mostly better connected than the Baan model, but that the latter frequently and rather spuriously includes a few key entities (such as product, product group, location, price, pricing categories) in most tables, artificially inflating its complexity scores. Low connectivity and fan-out values are, however, useful indicators for shallow models e.g. the relationships in the BelgAcc have been modelled incompletely and the Inmon model is a very shallow model designed from a data warehousing perspective. The inadequacy of both the average and standard deviation of fan-out is discussed further below.

- The absolute and relative data bang metrics are of little or no interpretative value. This is illustrated when comparing models from the same reference discipline and similar complexity, e.g. TOVE rates almost 20 times more absolute data bang than AIAI, though it is not nearly that much more complex. Hay and Silverston are roughly comparable models in terms of size and complexity but Hay has 2 to 2 ½ times as much data bang as Silverston. SAP scores less than Baan on both measures though its model is better connected. Inmon's score should be less than BOMA or Fowler.

The above concerns are not intended to be a full motivation, because that would depend on a more detailed discussion of the complexity and quality of the various models in the test bed and a lengthy analysis of the metric rankings for each model. The critiques are obviously also not meant to imply anything about the usefulness of the respective metrics for *software* design measurement.

7.5 Visualizing the Model Networks and Syntactic Model Signature[5]

Applying classical software engineering metrics to the field of high-level (business) domain modelling was not very successful for the purposes of model analysis. What is needed is a minimal set of metrics which enables the user to visualize the model network intuitively, including model size, structure and connectedness. The following documents the quest for the ideal measure, which looked at a classic network visualization approach such as plotting the models as networks, graphing the entity fan-out distributions and calculating the descriptive statistics of these distributions.

[5] Part of this section was presented at the SAICSIT 2002 conference and published as [VANB02]

7.5.1 Drawing the Model Network Using Graph Analysis Packages

Graph analysis researchers have developed a number of tools to help with graph analysis and visualization. Some of the more popular tools are the following [www.google.com]:

- Graphviz: open source graph drawing software from AT&T and Lucent Bell Labs
- AGD: Algorithms for Graph Drawing from a German collaborative group.
- Jviews Component Suit from ILOG, a commercial package.
- GDT: Graph Drawing Toolkit from the DIA lab at the University of Rome.
- PAJEK: Package for Large Network Analysis from the Vlado group at the University of Ljubljana, Slovenia.
- AiSee Graph Layout Software: another commercial package.
- PIGALE: Public Implementation of a Graph Algorithm Library and Editor, a free library of C++ routines.
- Tulip: also released under General Public License.

For purposes of visualizing the model networks, PAJEK was found to be user-friendly yet it still has a comprehensive functionality and generates graph diagrams quickly using many different layout criteria. It appears to be used quite commonly in graph research. Additional benefits are that it is available free of charge and runs on a Windows platform.

Of the many options and network manipulations available, the following four diagrams appeared to be most productive because they produce charts that were quite distinctive and did not require the input of arbitrary parameters:

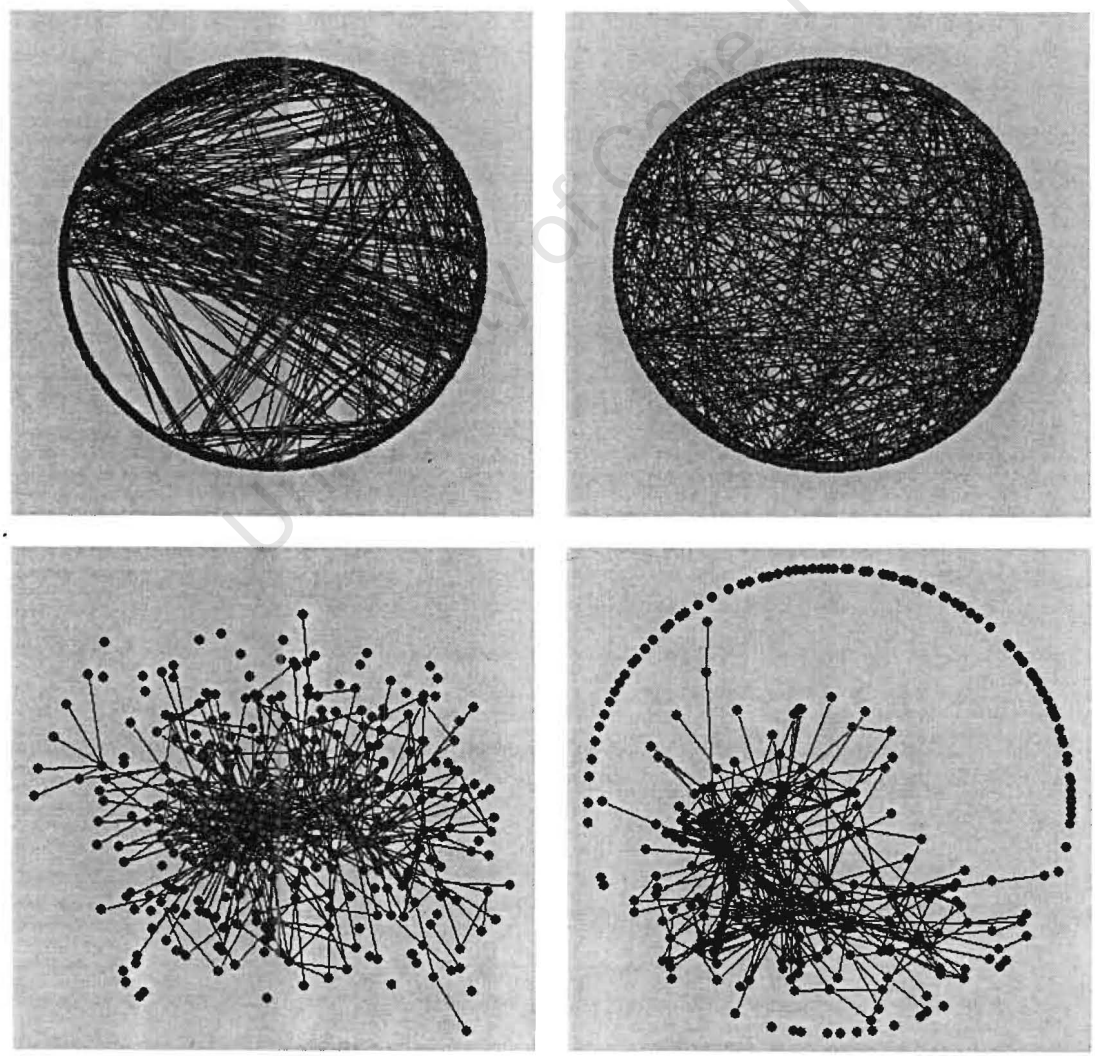


Figure 7-1: Network Plots For SAP Using Pajek.

- A plot showing entities in a circle in the sequence in which they occurred in the original source and relationships by connecting lines (Figure 7-1, top left).
- A circular plot as above, but with the entities re-arranged in random order (Figure 7-1,top right).
- A plot with the entities positioned according to the Kamada-Kawai algorithm. This algorithm plots the entities (little circles) on the graph so that the geometric (Euclidean) distance between them is as close as possible to the graph-theoretic (path) distance between them. It is an attempt to redraw the (n-1)-dimensional network onto two dimensions (Figure 7-1, bottom left).
- A plot using the Fruchterman-Reingold graph layout algorithm. This is an "energy positioning" iterative procedure whereby the entities that are connected attract each other and unrelated entities repel each other (Figure 7-1, bottom right).

Pseudo-code used for the Kamada-Kawai and Fruchterman-Reingold algorithms can be found at http://repast.sourceforge.net/docs/api/uchicago/src/sim/gui/LayoutWithDisplay.html.

Figure 7-1 illustrates the four diagrams for the SAP model. In order to save space, not all plots have been included. Appendix L gives examples of all four plots for those models that are similar in size: SAP, Baan, Hay, Inmon, Silverston and Random. These models were drawn using only the "domain relationship" and ignoring any structural "Is-A" relationships. Adding the latter relationships makes the diagrams even less distinct between models.

It was found that the "look and feel" of the resultant diagrams was heavily dependent on the number of nodes (entities). To illustrate the influence of the number of entities on the plot, one of the smaller models, AIAI, is also included in the appendix.

It is dangerous to rely on the circular plot of the entities in original (input) order. There is a pronounced tendency for entities that appear together in the same group or diagram to be much more closely related, and thus they tend to have the highest density of relationships between them. This means that most of the relationships are between nodes (entities) that are entered closely together. When plotted, this results in connections between entities (dots) occurring in fairly adjacent positions at the edge of the circle, which are hidden from view since they do not cross the circle surface. The first diagram therefore is more representative of the number of relationships *between* groups or diagrams. An extreme example of this phenomenon is the plot of the Inmon model (Figure 7-2) which appears extremely shallow, until it is realized that the bulk of the lines (connectors) actually occur on the rim.

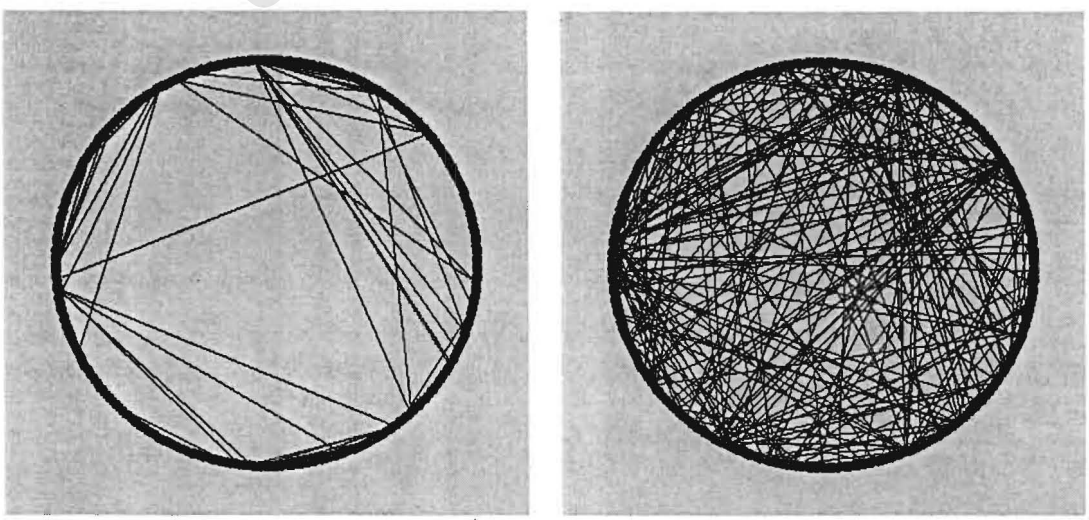


Figure 7-2: Ordered versus Random Circular Plot for the Inmon Model .

Consequently, a graph with entities reorganized in random order gives a better view of the overall density of the network since the lines are more evenly spread out. The more lines, the denser the network is on average.

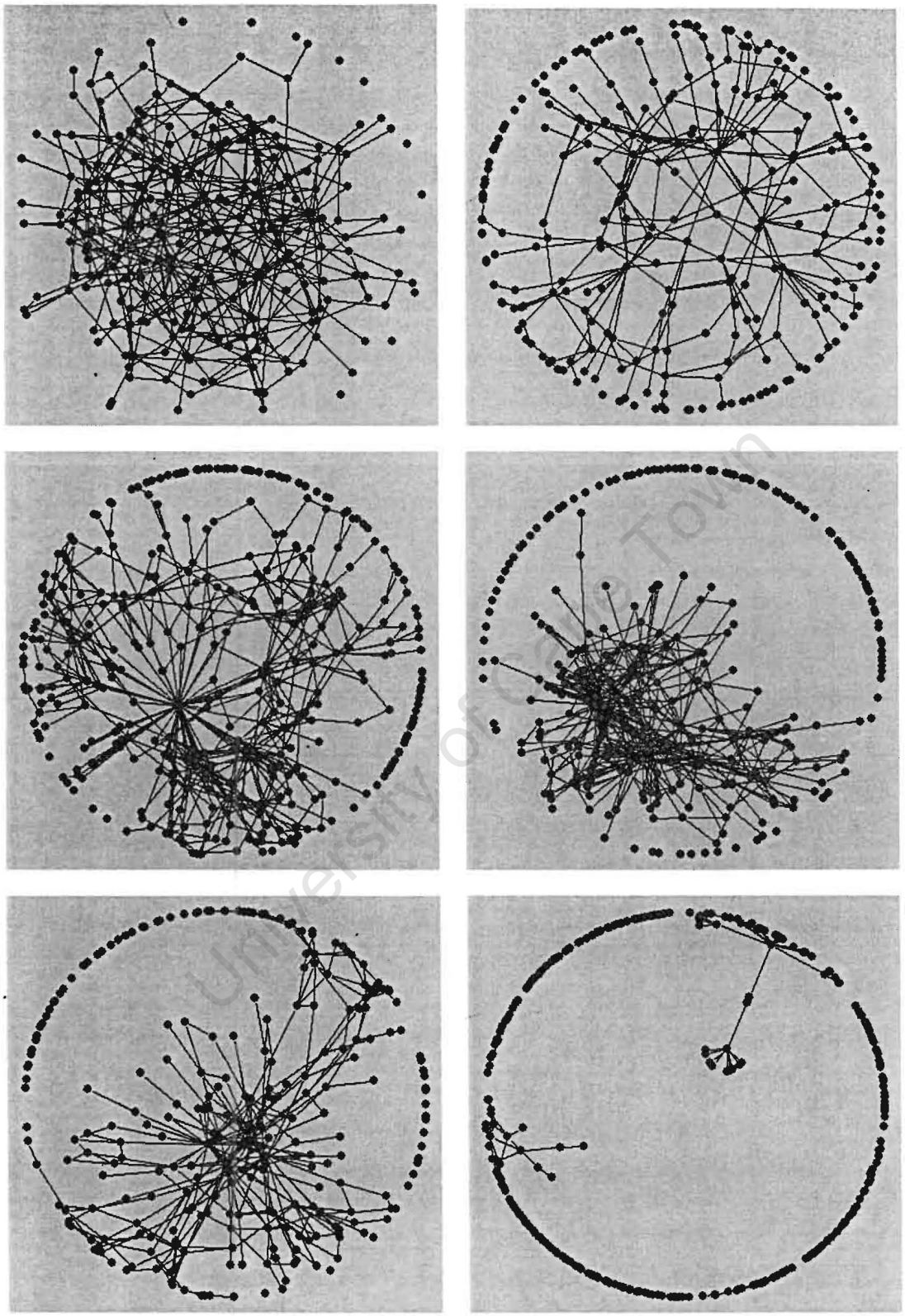


Figure 7-3: Comparative Fruchterman-Reingold Graphs for Similar Sized Models.

The more sophisticated diagrams do show some distinctive differences but they are hard to interpret in a systematic way. One distinctive feature suggests itself when comparing models of similar size and that relates to the relative spread or clustering of relationships. Where groups of entities are highly

clustered, with relatively few inter-cluster relationships, the Fruchterman-Reingold graph shows more open space. The relative spread of inter-relationship clusters is clearly visible in the clustering of the lines of the graph. Figure 7-3 gives 6 prototypical plots along the continuum from no clustering to extreme clustering.

- The random model (top left) is the prototype of no clustering of relationships, except by random assignment. The relationships are spread out evenly, and any entity is equally likely to be connected to any other entity.
- The Inmon model (bottom right) is by far the most clustered model within the model database. Entities appearing on the same diagram are linked to each other (these appear in sequence clusters on the rim) but very few entities have links to entities on other diagrams. The diagram is mainly empty apart from one or two entity groups which have been pushed to the inside.
- Silverston (top right) is an example of a model with relatively little clustering, i.e. a fairly even distribution of relationships with entities within the model not closely clustered.
- SAP (middle left) shows evidence of a number of central concepts which keep reappearing on different diagrams ("time", "product" etc.) as evidenced by the few tight "bundles" or concentrations of lines on the diagram.
- The Baan model (middle right) and Hay (bottom left) display an increasing "cluster" tendency i.e. the entities are not spaced evenly but closely nested together.

Apart from the above observations, it appears that graphs of models with comparative complexity did not appear to have any distinctive features that could serve as discriminating qualities. It became even more problematic to distinguish patterns that persisted between models of different sizes.

7.5.2 Plotting the Fan-out Distributions

Combining the approaches from the graph analysis (plotting entities and relationships as a network) and the more traditional system engineering metrics (especially the fan-out and data-bang algorithms), led to the idea of plotting the frequency distributions of the fan-outs for each model.

To reiterate, the fan-out of an entity is the number of relationships in which a given entity participates. Using network terminology this is the number of arcs connecting a given node. A very "light" model, i.e. a model with few relationships relative to the number of entities, will have a proportionally large concentration of entities with low fan-outs. For a complex, strongly interconnected model, the distribution of fan-outs will move left with a relatively high number of high fan-out entities.

The inheritance relationships have been excluded from this analysis. It is in principle possible to repeat the distributional analysis as below on the inheritance relationships only (using the NOC / Number Of Children metric for the nodes which are part of the inheritance tree) or on the conflated models i.e. incorporating both the domain relationships and the structural inheritance relationships. However, it is felt that this would add little value to current model analysis, except where models with very detailed inheritance structures need to be compared.

A characteristic of the fan-out distributions is the large proportion of outliers: most models have a small number of high fan-out entities: entities that have a large number of relationships such as product (or part), customer or employee. Table 7-7 illustrates that the entities with fan-outs exceeding 9 account for a relatively small proportion of the distributions for virtually all of the models.

Table 7-7: Entities with Fan-outs Exceeding 9.

Fan-out	Nr of models containing a node with this fan-out	Total number of nodes with a greater fan-out	% nodes with greater fan-out
9	20	259	5.4%
10	17	200	4.2%
11	14	167	3.5%
12	14	129	2.7%
13	10	112	2.3%
14	9	98	2.1%
15	10	80	1.7%
16	5	72	1.5%
17	4	67	1.4%
18	4	60	1.3%
19	3	57	1.2%
20	4	53	1.1%

Table 7-8 summarizes the fan-out distributions for each individual model.

Table 7-8: Relative Fan-out Distributions for All Models.

Fan-out	1	2	3	4	5	6	7	8	9	10+	Total
Purdue	1%	15%	19%	15%	15%	9%	13%	3%	4%	6%	100%
Miller	12%	0%	35%	4%	19%	0%	0%	4%	4%	23%	100%
Random	9%	18%	22%	18%	13%	10%	5%	3%	1%	0%	100%
Semi-Random	9%	18%	22%	18%	13%	10%	5%	3%	1%	0%	100%
Hay	20%	22%	13%	8%	7%	6%	3%	4%	2%	16%	100%
TOVE	19%	20%	19%	14%	10%	5%	3%	2%	2%	6%	100%
SAP	15%	32%	22%	10%	7%	5%	3%	2%	1%	5%	100%
Ottawa-Dense	10%	37%	19%	11%	6%	6%	2%	2%	1%	5%	100%
Baan	27%	15%	11%	11%	5%	4%	6%	3%	3%	16%	100%
USB	24%	14%	14%	10%	10%	10%	3%	7%	2%	6%	100%
Silverston	33%	33%	15%	3%	3%	3%	3%	1%	2%	4%	100%
CYC	29%	23%	17%	10%	5%	5%	2%	1%	1%	6%	100%
SanFran	30%	21%	15%	7%	3%	7%	3%	5%	5%	3%	100%
AIAI	34%	27%	5%	16%	0%	5%	2%	0%	2%	9%	100%
ARRI	36%	21%	7%	9%	7%	5%	6%	3%	3%	3%	100%
BOMA	33%	24%	18%	7%	5%	4%	2%	2%	0%	5%	100%
Ottawa-Big	44%	23%	12%	7%	4%	3%	3%	1%	1%	4%	100%
Fowler	36%	27%	18%	3%	4%	1%	3%	2%	2%	3%	100%
NHS	46%	27%	14%	4%	3%	2%	1%	1%	1%	2%	100%
AKMA	43%	30%	13%	7%	2%	2%	2%	0%	2%	0%	100%
BelgAcc	54%	22%	10%	6%	2%	1%	1%	1%	0%	3%	100%
Inmon	75%	7%	5%	1%	3%	1%	1%	1%	1%	4%	100%
Nippon	52%	26%	7%	10%	2%	0%	2%	2%	0%	0%	100%
average	**30%**	**22%**	**15%**	**9%**	**6%**	**5%**	**3%**	**2%**	**2%**	**6%**	**100%**

The distributions are much easier to visualize using a ribbon plot as shown in Figure 7-4. It is clear that the distributions are not typical bell-shaped distributions, but severely skewed. Hence a relatively large difference between the various measures for "average" (median, mode and mean) can be expected. In an attempt to group similar distributions close to each other, the models have been ordered from the highest median (representing the 50% "centre" of the frequency) to the lowest.

Table 7-9: Descriptive Statistical Measures for the Fan-out Distributions.

Descriptive Statistic	N	Arithmetic Mean	Mode	Median	Harmonic Mean	Std. Dev.	Range	Pearson Skewness1	Pearson Skewness2	3rd Adj. Mom.	Curtosis Coeff	4th Adj. Mom.
Purdue	89	5.03	3	5	3.82	2.68	15	0.04	0.76	1.24	0.29	1.98
Miller	26	6.23	3	4.5	3.28	5.12	21	1.01	0.63	1.39	0.33	1.17
Random	238	3.82	3	4	2.80	1.91	10	-0.28	0.43	0.63	0.38	-0.04
Semi-Random	238	3.82	3	4	2.80	1.91	10	-0.28	0.43	0.63	0.38	-0.04
Hay	235	6.17	2	3	2.42	9.12	73	1.04	0.46	3.93	0.22	18.88
TOVE	539	4.51	2	3	2.33	7.81	127	0.58	0.32	10.24	0.21	138.21
SAP	328	3.73	2	3	2.30	4.89	73	0.45	0.35	9.41	0.17	122.35
Ottawa-Dense	248	3.67	2	3	2.45	3.04	22	0.66	0.55	2.79	0.20	9.85
Baan	232	5.24	1	3	2.23	5.94	43	1.13	0.71	2.81	0.33	10.52
USB	119	4.02	1	3	2.29	2.80	12	1.09	1.08	0.85	0.29	-0.05
Silverston	209	3.08	1.5	2	1.76	3.60	26	0.90	0.44	3.77	0.17	17.48
CYC	639	3.60	1	2	1.94	5.48	105	0.87	0.47	11.50	0.25	190.72
SanFran	99	3.47	1	2	1.95	2.84	12	1.56	0.87	1.28	0.29	0.84
AIAI	44	3.32	1	2	1.81	3.09	12	1.28	0.75	1.64	0.19	1.66
ARRI	119	3.31	1	2	1.81	2.85	15	1.38	0.81	1.58	0.33	2.63
BOMA	128	3.00	1	2	1.81	2.58	15	1.16	0.78	2.03	0.30	4.43
Ottawa-Big	459	2.76	1	2	1.59	2.99	24	0.76	0.59	3.31	0.20	14.02
Fowler	95	2.76	1	2	1.71	2.37	12	0.96	0.74	1.93	0.17	3.30
NHS	173	2.54	1	2	1.52	3.56	35	0.46	0.43	6.03	0.33	45.11
AKMA	56	2.18	1	2	1.54	1.60	9	0.33	0.73	2.15	0.33	5.28
BelgAcc	178	2.39	1	1	1.41	4.43	54	0.94	0.31	9.33	0.17	101.88
Inmon	225	2.14	1	1	1.22	2.99	23	1.14	0.38	3.95	0.00	19.34
Nippon	58	2.00	1	1	1.42	1.49	8	2.02	0.67	2.05	0.17	4.55
average	**208**	**3.60**	**1.54**	**2.54**	**2.09**	**3.70**	**33**	**0.84**	**0.60**	**3.67**	**0.25**	**31.05**

7.5.3.1 Measures of Central Location (Averages)

The table lists the most common descriptive statistical measures to indicate central location (averages): the arithmetic mean (sum of fan-outs divided by number of entities), the mode (the fan-out with the highest frequency), the median (the fan-out for the middle point of the distribution i.e. the 50%-point on the cumulative frequency distribution), and the harmonic mean (the n^{th} root of the product of the fan-outs).

As can be expected from unimodal distributions, the median generally lies between the mean and the mode. Because of the very discrete nature of the distribution, the relationship does not always hold. In the statistical analysis of frequency distributions, it is common practice to interpolate the mode and medians between the integer values, especially where the data is summarized class data and the original detail data is unknown. However, this is not recommended in this case because of the following reasons:

- The data is *not* class data (apart from the 10+ category).
- There is no conceptual meaning: the semantics of the fan-out metric dictate that it must be an integer value since an entity cannot have, say, 2.24 relationships.
- No additional insight results from fractional values.

The only argument in favour of calculating fractional values for median and mode would be the usefulness in terms of ranking. Because of the irregularly shaped distributions of the surveyed models, this does not apply in our case.

From an intuitive and admittedly subjective interpretation of the chart, it appears that the harmonic mean gives the best feel for how the data should be ordered: it summarizes the ranking given by the

combination of arithmetic mean, mode and median in one single measurement, regardless of the shape of the distribution. A more objective explanation for the reason why it should be preferable over the arithmetic mean (as used to calculate the "average fan-out" metric as proposed in the software engineering discipline) is the fact that it tends to downplay the effect of outliers and fat tails which characterize the fan-out distributions but, unlike the mode and median, still takes the entire distribution into account.

7.5.3.2 Measures of Dispersion

To measure the dispersion of fan-outs, two measures are commonly used. The range gives the absolute width of the fan-out frequency distribution. Since all models have at least one entity with a fan-out of 1, the range for all models also happens to equal their maximum fan-out value. The expected (or average) distance of actual fan-outs away from the average is given by the standard deviation.

From Table 7-9, it is clear that there is a fair degree of correlation between range and standard deviation. However, when these measures are compared against the chart, the measures of dispersion appear to give little guidance for the overall look and feel of the fan-out distributions. For instance, the BelgAcc or NHS models have relatively high values in comparison to the Random model despite the fact that the latter looks more dispersed. It appears that both dispersion measures are fairly susceptible to the outliers (high fan-out entities), as is well documented in statistics. Another problem is that the dispersion measures cannot be compared well across the different types of curves, e.g. the standard deviation for a typical "wave" does not compare well against a typical "slide" (concepts to be explained in 7.6.1.2).

7.5.3.3 Skewness

Table 7-9 and Figure 7-4 clearly show the skewed nature of the distributions. Two main statistical measures of skewness are commonly proposed [KANE68]:

- The easy to calculate Pearsonian skewness coefficients: the first Pearson coefficient is based on the standardized difference between arithmetic mean and median, the second one is based on the difference between mean and mode.
- The more mathematically based adjusted third moment i.e. the third moment corrected for the ordinary dispersion of the underlying distribution. This is an extension of the calculation of arithmetic mean (related to a first moment, using a power exponent of 1) and standard deviation (second moment, using power exponents of 2).

Kane warns that the interpretation is not straightforward: non-symmetrical distributions can have third moment values of close to zero, and symmetrical ones can have relatively high values.

In our case, the third moments of all distributions exceed one half in absolute value, which [KANE68] regards as the critical value above which a distribution is "noticeably skewed". However, the values themselves do not convey much additional information (such as ranking) about the distributions, nor do they correlate well with the Pearsonian skewness coefficients. The latter do not even correlate well among themselves, and appear to add no value whatsoever to the description of the distributions. The lack of interpretative value is probably due to the fact that the skewness measures are really meant for continuous distributions and do not work well given the strong discreteness at the lower end of the distributions, where the median and mode are located.

7.5.3.4 Peakedness or kurtosis

Kurtosis – or peakedness – relates to an even higher-order set of descriptive statistics. These are of an even more problematic interpretation value, even with more regular and continuous distributions. There are claimed to be three main types of kurtosis: leptokurtic (sharp or steep), platykurtic (flat) and mesokurtic, but these interpretations are applicable only to bell-shaped distributions. Again, two measures are suggested in the literature:

- The algebraically easily interpretable "coefficient of kurtosis", measured as the ratio of the semi-interquartile and 90/10 percentile ranges; and
- The mathematically more complex adjusted fourth moment i.e. the third moment divided by the fourth power of the dispersion minus the constant 3 (to normalize it for the normal distribution).

There appears to be even less correlation between the two alternative measures than was the case for the skewness measures. This may again be attributable to the same reasons as discussed for the skewness measures: the discreteness and irregular shape of the distributions. Note in particular how the 4^{th} moment is dramatically influenced by the maximum fan-out (as indicated by the range) since it is raised to the power 4.

7.6 The Proposed Fan-out Frequency Distribution Characteristics

The "traditional" descriptive frequency distribution statistics also proved to be not very adequate for the type of distributions associated with fan-out frequencies. This is due to the distinct irregular shapes of the distributions: extremely skewed with a mode of 1 or 2 and the large number of outlier values. The following discusses the proposed statistics to characterize the fan-out distributions.

7.6.1 Introduction of Proposed Model Signature

7.6.1.1 Plotting the Fan-out Frequency Distributions.

After experimentation with various two- and three-dimensional graphing formats, it was found that a *three-dimensional line* chart facilitates visualization and classification of the distributional shape, as illustrated in Figure 7-4. Fan-outs larger than 9 have been omitted from the chart.

7.6.1.2 Subjective Description of the Shape of the Fan-out Distributions

It is clear from the figure that the most model fan-out frequency distributions are not very "normally" shaped i.e. the traditional bell curve does not apply very well. Figure 7-4 suggests a small number of distribution or *curve shape families*, which might be characterized as follows:

- ***Waves***: increase or build up sharply from a low value to a peak value and then drop back slowly to the base level. The prototypical example is the TOVE model. Some waves, such as SAP or Ottawa-Dense are extremely peaked ("tidal waves"?), whereas others are much smoother. A formal definition for a wave would be that the mode exceeds the value of one.
- Slides (or skateboarding slopes?): fall sharply from a high value to the base, with a "soft landing" whereby the curvature reduces as one gets closer to the base. The BelgAcc model is a prototypical example. These are basically waves without peak or build-up phase, i.e. their mode equals (a fan-out of) one.
- All distributions vary in degree of smoothness: TOVE and BelgAcc are relatively smooth and USB, SanFran and Miller are very bumpy curves.

- The extremely bumpy curves could possibly be reclassified in a separate category, the "rollercoasters", which would include Purdue, Miller, USB and SanFan.

Note that Silverston's shape does not neatly fall into the conceptual class of waves or slides. Here an arbitrary assignment to the wave or slide category could be made for distributions with a mode of 1.5.

The only difference between the slide and wave (with mode 2) is the number of entities with a fan-out = 1. A qualitative or normative interpretation of the metric would naturally lead to an investigation of the nature of these entities in the "slide" models.

7.6.1.3 Bumpiness and Smoothness

As mentioned previously, one of the prime characteristics of the distributions is the *relative bumpiness or smoothness* of the distributions. Conceptually, bumps (and troughs or valleys) are measured by local maxima or minima in the curve; these can be found by calculating the first differentials. However, subjectively, it is really the number of inflection points which contribute to the visual effect of bumps (and troughs). These can be measured by the second differential: a change in curvature (from increase to decrease or vice versa) happens whenever the second differential changes sign.

These differentials can be calculated very easily. Following the same argument as for Figure 7-4, the calculation is limited to the first nine points of each distribution. Applying the calculations to all data points introduces many additional irrelevant inflection points which are not really indicative of the overall shape of the curve. Thus the traditional statistical practice of ignoring outliers has been applied here in the interest of finding a measure to characterize the most important part of the distributions. These outlying inflection points disappear anyway when a sensitivity threshold value is introduced.

The first "bumpiness" calculation was based on the calculations of any change in sign. Consequently, this measure includes even the most minute inflection points. The resultant bumpiness score has a minimum of 0 and a theoretical maximum of 6: since there are 9 original values, there are 8 first differences and 7 second differences, implying a theoretical maximum of 6 sign reversals.

A more practical approach is to only consider "noticeable" inflection points: the human eye cannot discern small bumps, and a tiny variation or deviation should be allowed. This entails applying some sensitivity threshold below which reversals are ignored. After some experimentation with various cut-off values and algorithms the following approach was found to mesh well with the subjective graph interpretation.

- There are basically two choices for calculating the threshold: adding versus multiplying (the absolute value of) the two differentials. A multiplicative effect seems to be a more logical choice when considering that the total effect becomes nil if any one of the two factors (second differentials) is zero.

- The cut-off value has to be set at a certain constant. By trial and error, a value of 2.5% was found to be a good cut-off point.

An attempt to find a more objective way to justify the cut-off point resulted in a different criterion. First, a "mid-value" bumpiness (2.5), halfway between the highest (5) and lowest bumpiness (0) factor, was determined. Then, using a goal or target-seeking approach, a cut-off value was calculated so that the average bumpiness value for all the models would equal this number, i.e., on average, the bumpiness of the models is halfway between the extremes! This value turned out to be 2.4% i.e. only where the product of the second differentials of the fan-out frequencies falls below –2.4% is a sign

reversal considered a "bump". In this case, the average bumpiness is 2.52, which is right in the middle of the maximum (5) and minimum (0) bumpiness factors for all models (the discreteness of the bumpiness values, makes it impossible to obtain 2.5 exactly).

Table 7-10 lists the bumpiness values. Comparing the "Bumpiness 2.4%" values from the table with the shape of the model fan-out distributions in the chart shows the intuitive appeal of the proposed metric. The very "bumpy" curves of the Purdue, Miller, SanFran models all merit their rating of 5, with the USB, AIAI and Fowler following closely behind in bumpiness (rating a 4). TOVE, Ottawa-Big and NHS equally merit their 0 rating due to their clearly visible smoothness, with the other models falling neatly in between. The merit of using the cut-off value is vindicated when one considers the cases of the Silverston and CYC models whose bumpiness fell from the extreme values of 5 and 4 respectively (0% cut-off), to their more deserved 1 rating with a 2.4% cut-off. It is felt that this metric deals very satisfactorily with the bumpiness/smoothness features, as well as in identifying the so-called roller-coaster curves.

Table 7-10: Proposed Syntactic Model Signature Metrics.

	Harmonic Mean	Bumpi-ness (0%)	Bumpiness (2.4%)	Slope of line fit	h coeff Tanh(x/h)	SSD
Purdue	3.82	5	5	-0.6%	0.67	6%
Miller	3.28	5	5	-1.5%	0.76	5%
Random	2.80	3	2	-2.0%	0.49	4%
Semi-Random	2.80	3	2	-2.0%	0.49	4%
Hay	2.42	4	3	-2.4%	0.58	2%
TOVE	2.33	1	0	-2.8%	0.46	0%
SAP	2.30	2	1	-3.1%	0.40	1%
Ottawa-Dense	2.45	4	3	-3.0%	0.42	2%
Baan	2.23	4	3	-2.5%	0.58	3%
USB	2.29	4	4	-2.2%	0.52	0%
Silverston	1.76	5	1	-4.0%	0.30	2%
CYC	1.94	4	1	-3.5%	0.38	1%
SanFran	1.95	5	5	-2.9%	0.42	2%
AIAI	1.81	4	4	-3.8%	0.36	5%
ARRI	1.81	4	3	-3.2%	0.39	3%
BOMA	1.81	5	3	-3.9%	0.33	1%
Ottawa-Big	1.59	2	0	-4.3%	0.27	2%
Fowler	1.71	5	4	-4.0%	0.29	2%
NHS	1.52	1	0	-4.7%	0.22	1%
AKMA	1.54	2	2	-4.7%	0.22	0%
BelgAcc	1.41	1	0	-5.1%	0.19	1%
Inmon	1.22	5	4	-5.4%	0.12	7%
Nippon	1.42	3	3	-5.0%	0.20	1%
average	2.09	3.52	2.52	-3.3%	0.39	3%

7.6.1.4 Shape curvature

Not having found a satisfactory higher-order statistical measure to summarize the overall shape of the distributions, the search is still on for a suitable statistic. It was noted earlier that the subjective classification into waves and slides could be defined objectively by means of the mode (i.e. waves have a mode > 1). If the bumpiness is ignored, the main remaining feature that distinguishes slides from each other, is the steepness, "bent-ness" or overall curvature (shape) of the slides; and a similar argument applies to the waves.

After considering many approaches, it was found that a linear regression analysis provided a rough approximation for the subjective curvature seen in each of the distributions. In particular, the calculated slope coefficient of the least squares straight line for each distribution correlates well with

the observed steepness. (Refer to the "slope of line fit" column in Table 7-10.) For example, the BelgAcc, Inmon and Nippon can all be considered to be much steeper than the Random, Hay and SAP models. However, the finer distinctions are not mapped too well: Inmon is much more curved than either Nippon or BelgAcc, but this difference is not very pronounced in the difference of slope values. The search for a better proxy i.e. a metric with a much finer granularity is still on.

The approach of fitting a curve using the least squares approach seemed to be a promising one. Consequently, a large variety of curves was tried: including hyperbolic curves, power curves, exponential curves and logarithmic curves. None proved to be very satisfactory. The main reason was that most curves have more than one parameter, and that there generally did not seem to be any discernable or constant pattern in any single parameter to isolate a suitable candidate metric.

As an illustrative example of the problem, consider the following power functions which are the best fits for four roughly similar distributions.

Model	Best-fit Power Functions
BelgAcc	$y = 1.6642\ x^{-3.0}$
Inmon	$y = 0.4056\ x^{-1.8629}$
Nippon	$y = 1.4480\ x^{-3.1939}$
AKMA	$y = 1.1108\ x^{-2.579}$

Although the full analysis for all models reveals an overall trend to smaller (absolute value) powers as the curves grew flatter, the fluctuation in actual values for closely related curves was too large for the metric to be of any use.

After a long process and extensive literature survey, an alternative approach was found in [John 1998] who used a "tanh" (hyperbolic tangent) function to fit a cumulative distribution curve. The tanh function can be defined as:

$$Y = \tanh(x/h) = (1 - e^{-2x/h}) / (1 + e^{-2x/h})$$

Where h is the shaping constant of the curve.

The beauty of this curve is that there is only one parameter to estimate which precludes any trade-off between multiple parameters to define closely related shapes.

Although there is no theoretical basis whatsoever for the use of this particular function, subsequent analysis showed that the tanh function indeed not only makes a beautiful fit between the actual cumulative distributions and the best-fit curve, but the plot of the cumulative distributions and the plot of the best-fit tanh functions show visually very clearly both the appropriateness of the model fit and the close match between the ranking of the model actual and fitted distributions. Again, it must be stressed that the function has been chosen purely on pragmatic grounds i.e. its visual fit and single parameter. No theoretical explanation seems to present itself as to why this curve fits so well and it is a matter for further research whether it would fit models from other domains equally well.

Figure 7-5 shows some sample comparisons for a number of the more extreme cases, including the difficult and "ill-behaved", bumpy Miller model. It must be stressed again that the purpose is not to find a perfectly fitting curve but to find a single-parameter curve which expresses the overall curvature of the distribution best. A chart containing the plots of the cumulative fan-out distributions and best-fit tanh functions for all the models is included in Appendix D.

As an interesting by-product, the sum of the squared differences (between actual and best fit functions) for each model (or individual distribution) was generated. As can be expected, there is indeed a good correspondence with the bumpiness factors as calculated earlier, underlining both the validity of the proposed tanh function fit as well as the bumpiness factors: if the tanh function is a

good approximation, the remaining differences can be explained by the bumps or lack of smoothness of the distribution.

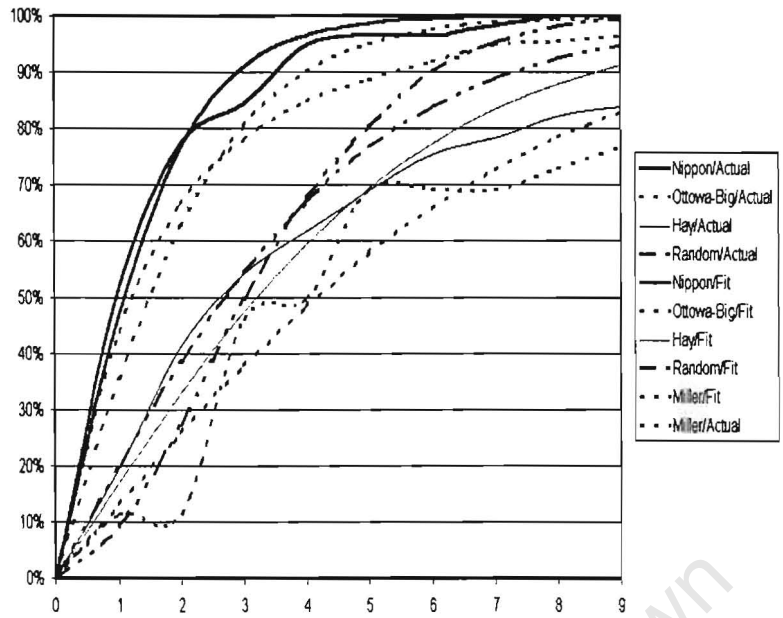


Figure 7-5: Comparison of Actual (Cumulative) Fan-out Distribution and Best Fit Tanh Function.

7.6.2 Conclusion

Traditional descriptive statistics and software engineering metrics proved inadequate in characterizing the frequency distributions of entity fan-outs. The following alternative metrics were proposed instead:

- An intuitive categorization into two or three types of curves: waves and slides, depending on whether the modal fan-out frequency exceeds one or not. Optionally, the term roller coaster can be used to denote very irregular curves.
- The best average or central location statistic is not the arithmetic mean, mode or median, but rather the harmonic mean, due to the presence of the outliers and the discrete nature of the lower values of the distributions, where the highest frequencies occur.
- A very satisfactory measure of overall bumpiness or smoothness of the distribution is found by looking at significant inflection points. These can be measured by sign reversals in the second differentials. The use of an appropriate threshold value ensures that spurious or very slight bumps are ignored; a method for calculating this threshold value was also proposed.
- The overall "bent-ness", steepness or curvature of the distribution proved to be a final characteristic metric. After a number of experimental explorations, it was empirically determined that the shaping parameter for the least-squares fit of a tanh function provided a very suitable metric.

Note that the proposed measures ignore the outliers completely. For completeness sake one other metric, such as the range or maximum fan-out which are both indicative of these outliers, could arguably be added to "complete the suite".

Also, the measures were intended to be purely descriptive i.e. to aid the visualization of the fan-out frequency distributions. Their intention was not normative nor evaluative, and different metrics would be necessary to evaluate the underlying models. For example, the selection of the specific formula

underlying the distribution curvature metric was purely pragmatic and had no theoretical foundation. The usefulness of the proposed measures for evaluating models would have to be derived from some theoretical model and validated within a wider modelling domain.

7.7 Architecture and Aesthetics.

When researching ways of measuring the quality of a model architecture based purely on objective, quantitative measures, some interesting approaches were found in the literature.

Although there is no clear definition of architecture in the context of models, the relevant part of the definition of software architecture applies: "composition of design elements; scaling and performance; and selection among design alternatives." This definition was quoted by [CLEM94] from [GARL93]. It is clear that many aspects of model architecture refer to the other syntactic criteria already investigated: size, structure, and complexity. In this section, the focus is on the compositional aspect of model architecture, namely the aesthetics of a model.

The approach which was followed has been suggested in [NGO00] and stems from research in the design layout of GUI interfaces. A second approach arises from construction architecture, and was an attempt to measure the emotional impact of a building by proposing quantitative measures for its temperature and harmony. This approach is discussed in Chapter 10.

A novel and exploratory way to analyse this aspect of model architecture is to look at the way the model diagrams are laid out. As discussed in Chapter 6, many models include diagrammatic model representations. There is a lot of literature available on interface layout [GALI97], although much of it takes the form of guidelines and principles rather than quantitative measures. One interesting, more quantitative approach is taken by Coleman in his quest to design an algorithm based on minimizing a composite function measuring 6 aesthetics-related graph layout aspects [COLE93; COLE96]. Although easy to implement, there are some methodological problems including the lack of explicit scaling of his metrics, no basis for determining the relative weights to be used for each factor and his focus on conceptual graphs which assumes that entities are represented by dimensionless dots and relationships by straight-line connectors. A similar approach was followed by [PURC01] who also attempted to provide some empirical validation for some of these aesthetics.

For this research, a more sophisticated set of 14 metrics as suggested by [NGO00] were used. They are an up-to-date, conceptually well-supported and comprehensive set of measures covering a wider range of aesthetic attributes of layouts. [NGO00] attempts to measure the ***aesthetics of a screen layout***, by decomposing the aesthetics into 13 distinct attributes or dimensions which they gleaned from the relevant literature. Formulae were suggested to quantify a layout's conformance to each of these dimensions, which can be summed to create a composite index. It was considered that, since many models are represented by means of diagrams, it would be worthwhile to consider if the formulae apply to model diagrams.

It must be understood that the metrics were designed specifically for interface layouts. A major problem with applying these measures to model diagrams is the fact that these also explicitly model relationships by means of connectors (of which there can be many different types of symbols on the same diagram, see e.g. UML). This introduces an important number of layout decisions including line cross-overs, number of lines, number of bends etc. It is suggested that, should the layout metrics be found to be fairly useful in the context of model diagrams, further research should be undertaken to extend the metrics to allow for these connectors.

An important pragmatic issue with the suggested metrics is that they involve fairly cumbersome measurements and calculations, especially if diagrams are not available in a standard vector format. A template was constructed to automate the calculations, but there is still a lot of effort involved in the measuring of position and size of each individual model element. The measures would be relatively easy to implement as part of a CASE or diagramming tool generating the model diagrams. Because of the exploratory nature of this research, a decision was taken to select only one single diagram for each model.

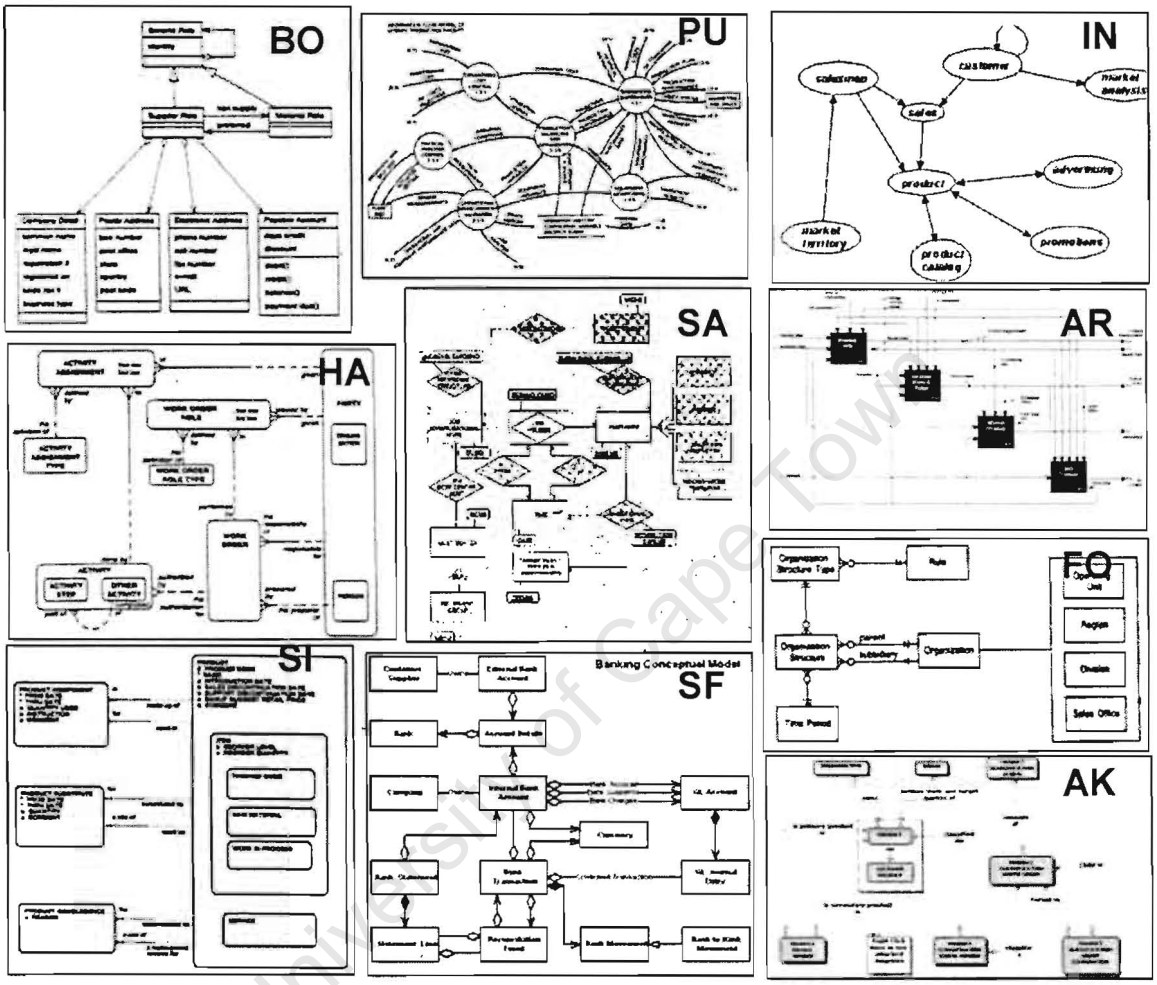


Figure 7-6: Representative Model Diagrams Used for Aesthetic Analysis.

Considerable care was taken to select a diagram that was considered to be as representative as possible for the diagrams used for the model i.e. in terms of size, layout, shape and complexity. Figure 7-6 gives a bird's eye overview of the diagrams selected to calculate the aesthetics metrics. Larger, more detailed printouts of the individual diagrams can be found in Appendix A. The 10 diagrams used as representative model layout samples are clockwise from top-right: Inmon (IN), ARRI (AR), Fowler (FO), AKMA (AK), SanFran (SF), Silverston (SI), Hay (HA), BOMA (BO) and Purdue (PU) with SAP (SA) in centre.

In calculating the metrics, the following simplifications were adopted for all diagrams:

- Three shades of grey were used: 0% (white); 100% (solid black) and 50% (anything in-between)
- Four shapes were recognized: rectangle (whether square or rounded corners); ellipse (including circles); diamond and triangles. Each shape is said to have a distinct aesthetic appeal.

- For the Inmon model, only a high-level diagram was used; its mid-level diagrams are distinctly different (and uglier?). For the SAP model, the attribute boxes and the ISA triangle were also included as diagram elements, since they contribute considerably to the clutter of the diagram. This was *not* done for the UML connectors.
- For the symmetry formulae, it is suggested that the actual angle (normalized to a range from 0 to 1 by dividing) be used instead of what amounts to the tangent of the angle in formula 17. Using the formula as given in the original article creates a severe distortion because angles close to the vertical axis have extremely high values. The corrected formula takes the ATAN of the ratio (to find the angle, expressed in radians) and multiplies this by $2/\pi$ (to yield a value between 0 and 1)

During the course of the calculations, the following problems were encountered, which concern the validity of the measures for model analysis and apply to all or most measures:

- The amount of text in a diagram is not taken into consideration.
- The distributions of many measures are too strongly grouped, often around the extremes of 0 or 1.
- There is no clear guidance on what to do with minute differences e.g. model elements that vary slightly in size but where the difference is hardly noticeably.
- A lot of measures rely on the allocation of model elements to a specific half or quadrant of the diagram. There is no guidance for model elements which lie exactly on a dividing line. The measures also do not account for objects whose centre may be slightly off a dividing line but where the surface area of the object is shared by a number of quadrants (or halves).
- Introducing colours (or different shades of grey) or different shapes to a diagram only has "penalizing" effects on the various measures; no measures "reward" the modeller for introducing diversity or distinction.
- There are no guidelines on what to do when a symbol is contained within another symbol as used in some diagrams to indicate grouper elements or super-types, for example.

The following are problems related to specific measures:

- The measures of balance and density do not allow for overlapping blocks e.g. UML "class-contains-subclass" notation. This penalizes diagrams such as Fowler, Silverston and Hay.
- Equilibrium: generally values very close to 1 are obtained.
- Sequence: does not allow for diagonal movement e.g. ARRI can be read perfectly in a diagonal line from top left to bottom right but gets "penalized" for leaving quadrants top-right and bottom-left empty.
- The way cohesion is measured is very questionable; alternatively it should receive a very low weighting. Does a landscape diagram printed on portrait paper really indicate lack of cohesion?
- The criterion of proportion is rather arbitrary in its selection of "ideal" shape ratios.
- Simplicity takes on very low values for any non-trivial diagram (i.e. containing more than a few objects) and is not very well distributed: a diagram of 1 object automatically has an SM of 1, a diagram with 2 objects cannot have an SM exceeding $3/(3+2) = 0.6$, the SM of a diagram with 3 objects cannot exceed $3(4+3) = 0.429$; the highest SM score for a 4 object diagram is $3/(4+4) = 0.375$ etc.

- Similarly, the measure of homogeneity is either 1 for a perfectly balanced diagram or very low (close to zero) for any diagram even slightly imbalanced. A diagram with a number of elements that is not a multiple of 4 is unfairly penalized. Elements that fall close to the centre lines are problematic because a move of a single element to another quadrant has a big impact on the overall homogeneity value.

The detailed formulae used to calculate the various aesthetics metrics are listed in Appendix C. In order to calculate the metrics, 30 different attributes were measured or calculated for each of the 117 model elements or objects found in the 10 diagrams. These were then used to derive 162 model attributes or model quadrant attributes, which along with a few model measurements, formed the basis for calculating the final 14 metrics.

Table 7-11 lists the values for each metric. The three models with the highest (best) score for each of the measures is highlighted by means of reverse text, and the worst three models are indicated by means of an underline.

Table 7-11: Aesthetics Metrics for 10 Model Diagrams.

	AKMA	ARRI	BOMA	Fowler	Hay	Inmon	Pur-due	SAP	San-Fran	Silver
Balance	71%	98%	73%	53%	65%	75%	90%	92%	58%	65%
Equilibrium	99%	100%	97%	97%	99%	100%	99%	100%	99%	98%
Symmetry	93%	87%	88%	91%	93%	95%	95%	97%	93%	91%
Sequence	33%	42%	42%	38%	8%	38%	33%	13%	25%	38%
Cohesion	78%	81%	69%	65%	51%	78%	78%	59%	78%	65%
Unity	43%	83%	71%	69%	64%	49%	86%	70%	53%	87%
Proportion	90%	87%	91%	96%	82%	90%	92%	84%	87%	85%
Simplicity	14%	25%	17%	14%	10%	12%	11%	4%	12%	13%
Density	85%	95%	62%	49%	64%	84%	90%	70%	72%	10%
Regularity	39%	42%	36%	53%	24%	6%	3%	46%	78%	36%
Economy	27%	100%	38%	60%	27%	38%	60%	13%	75%	38%
Homogeneity	22%	25%	13%	11%	4%	33%	22%	5%	0%	17%
Rhythm	96%	95%	95%	95%	97%	98%	98%	99%	96%	94%
Order/ Complexity	61%	74%	61%	61%	53%	61%	66%	58%	64%	57%

Note that the calculations were done using only the *entity* objects (areas) and not the *relationships* (lines) or any *text* found on the diagrams.

If one looks at the individual measures, the following observations can be made:

- In terms of balance, the symmetrical distribution of optical weight, ARRI and Purdue are clear leaders. Which are the worst balanced models depends very much on how the groupers or supertypes are counted, although the unused space in the top-right corner of the diagram (Figure 7-6) costs SanFran dearly.

- Equilibrium scores, measuring the centeredness of the entire diagram, are extremely high for all models, so this does not appear to be a problem for any diagram.

- It may appear surprising that both SAP and Inmon exhibit high symmetry, but the mathematical averaging out of the different model elements seems to work out very close to zero (as shown also in the equilibrium scores above). ARRI scores deservedly lowest because of its clear diagonal asymmetry.

- The sequence, the natural flow for the *Western eye* from top-left to bottom-right, is indeed best for the ARRI, although BOMA scores surprisingly high. On conceptual basis, ARRI should have

scored full marks but technical reasons (equal element sizes and empty quadrants) reduce its score dramatically. Hay deserves its low score, but that should have been shared by BOMA and, possibly, Silverston.

- Hay scores also low on cohesion, which is really calculated as a measure on how regular-sized the screen elements are. Hay and Silverston score very low due to their long boxes, SAP because of its irregular, almost squarish diagram size.
- Unity looks at the closeness and similarity of the various screen objects, and ARRI and Purdue score again quite high due to their relatively few, close, similar elements, although AKMA and Inmon also come close. Deservedly worst are Hay and Silverston with their unequal and widely spaced screen objects.
- Proportion looks at how aesthetically pleasing the screen shapes are i.e. how close they come to what are claimed to be universally pleasing shape dimension ratios such as the golden ratio (1:1618) or perfect squares. Getting a high score appears to be more a matter of luck than design, although their ugly, long rectangles send Silverston and Hay to bottom positions.
- In terms of simplicity, the very sparse ARRI and Purdue diagrams properly take honours, with the very cluttered SAP deservedly being regulated to the very bottom of the pile.
- Density measures to what extent the screen is covered with objects. Apparently, the less space is used, the higher the score. Examples are ARRI, Purdue, AKMA and Inmon. The very low score for Silverston is entirely technical, due to the fact that overlapping objects should not be added together, but the formula makes no provision for this eventuality.
- Regularity refers to the uniformity of the screen elements and looks at the number of different alignment points. It is indeed clear that the SanFran model uses a very matrix-like frame on which to hook its elements, followed at some distance by Fowler and SAP who also align their elements. Conversely, the irregularly plotted objects for Inmon and Purdue result in their very low scores.
- Economy refers to the number of different elements used and ARRI scores full marks because it uses only one single shape. Second position goes deservedly to both Fowler and Purdue, who use only two different shapes. SAP scores deservedly very low because of its many different shapes and sizes.
- Homogeneity measures how evenly the objects are distributed among the quadrants and is based on Bolzman's entropy formula. No model appears to do particularly well, although AKMA, Purdue, Inmon and, somewhat unexpectedly, ARRI receive high scores. SanFran scores a 0 due to its empty top-right quadrant though Hay's imbalanced distribution of objects also costs it dearly. Note that the nature of this measure's calculation makes it very sensitive to the objects lying close to dividing centre-lines.
- The rhythm looks at regular patterns i.e. how systematically the various screen objects are ordered vertically, horizontally or diagonally. Because of the large number (18) of constituent factors, differences are generally averaged out and this results in relatively high scores for all models, since all diagrams spread the objects fairly well across the diagram.

Overall average aesthetics score, called "*order/complexity*" in [NGO00]:

- Two models consistently score highest in most categories: the ARRI model and the Purdue model as evidence by the large number of black blocks in their column. Not surprisingly, their overall average therefore also puts them in first and second position. It appears that their simplistic, clean,

sparse and regular model layout implies high scores on many of the aesthetic layout measures. Following closely behind in third is SanFran.

- Two models that tie for worst layout overall, Hay and Silverston, indeed have consistently low scores for many of the individual measures. Coming third worst overall is BOMA. Indeed, the immediate impression for these models is one of some imbalance and irregularity.
- All the remaining models are clumped in the middle with identical 63% scores.

The design of many metrics was of such a nature that the measure could *theoretically* range from 0 to 1. In practice, no model diagram will, for instance, put all model elements in one quadrant of the diagram, or design a diagram with only one element. Unfortunately, some measures are more sensitive to pragmatic considerations than others, resulting in some scores having a very wide spread and others much lower spreads. This could, in principle affect the calculation of the overall "order/complexity" score quite dramatically. It must be mentioned that the original authors [NGO00] realize and acknowledge this, since they finish the article with the statement that "choosing weights is an unresolved issue". In the end, and in the absence of any empirical data, they suggested equal weights as was used above.

A practical approach is hereby suggested to calculate the weights differently, by trying to minimize the impact of measures that display an unusually wide or narrow range of scores. The suggested transformation is to re-normalized the scores in such a way that, for each individual measure, the lowest scoring model gets a 0% score and the highest scoring model 100%; i.e. each row undergoes a linear transformation which produces a range of exactly 100%. This does not change the relative ranking of any model but facilitates model discrimination by emphasizing the differences between models.

Table 7-12: Normalized Aesthetics Metrics for Model Diagrams.

Re-normalized	AKMA	ARRI	BOMA	Fowler	Hay	Inmon	Pur-due	SAP	San-Fran	Silver
Balance	41%	100%	45%	0%	28%	49%	82%	87%	13%	28%
Equilibrium	63%	99%	7%	0%	74%	100%	63%	96%	71%	43%
Symmetry	59%	0%	4%	37%	56%	73%	75%	100%	61%	32%
Sequence	75%	100%	100%	88%	0%	88%	75%	13%	50%	88%
Cohesion	90%	100%	61%	47%	0%	88%	90%	27%	89%	46%
Unity	0%	92%	63%	61%	48%	13%	99%	61%	23%	100%
Proportion	59%	33%	65%	100%	0%	61%	74%	16%	38%	25%
Simplicity	46%	100%	61%	46%	29%	36%	34%	0%	39%	43%
Density	89%	100%	62%	47%	63%	88%	95%	71%	73%	0%
Regularity	48%	51%	44%	67%	28%	4%	0%	57%	100%	44%
Economy	17%	100%	29%	54%	17%	29%	54%	0%	71%	29%
Homogeneity	66%	75%	37%	33%	10%	100%	66%	14%	0%	50%
Rhythm	50%	22%	14%	20%	59%	88%	80%	100%	40%	0%
Order/ Complexity	**54%**	**75%**	45%	**46%**	32%	**63%**	**68%**	**49%**	**51%**	41%

The normalization does not affect the analysis given above for any of the *individual* measures, although it does amount to a changing of the weights used to calculate the final order/complexity (bottom row), which now has different values for each diagram. A short analysis of the new values for the overall order/complexity – or aesthetic score – is in order.

The re-normalization (in effect a linear transformation) does not affect the top two positions, which remain Purdue and ARRI. However, Inmon now replaces SanFran in third position, the latter

dropping significantly. AKMA comes in quite a bit lower as fourth position. The re-normalization appears to agree with instinctive judgement: the new top four diagrams appear, at least to me, indeed the most balanced and pleasing to the eye.

Another significant effect takes place at the bottom: SAP's composite score no longer lies significantly lower than the middle group – which appears a justified move – and BOMA, with its irregular heavy bottom joins the bottom markers. The three new bottom diagrams are now indeed Hay, Silverston and BOMA although Fowler is also close!

Overall, it can be concluded that many of the suggested aesthetic measures do indeed provide an objective (and programmable) way to calculate the 13 proposed dimensions or aspects of aesthetic diagram layout. A suggestion was made on how to weigh the different components in a way that the final overall composite aesthetic score is not subject to the particularities of any one formula. Further research, however, is advised to address some of the shortcomings of applying the formulae design for screen layout to model diagrams, as mentioned under the general and more specific validity comments above.

7.8 Summary and Validation of Syntactic Measures

The syntactic model analysis is concerned with the purely structural aspects of the model, regardless of the underlying meaning of the model and its elements. Most of the analysis techniques are derived from the hard sciences i.e. the disciplines of software engineering, computer science, graph theory and network analysis. This includes a large variety of standard syntactic metrics relating to size, grouping, layering, inheritance structure, and network visualization, as well as some less standard metrics such as interface aesthetics.

Three ***size*** measures were used: entity count, CASE size (or concept count) and the suggested adjusted concept count. All have a fairly high face, content and construct validity. Criterion validity was attempted by looking at the model capture time but, although there was a fair degree of correlation, this was not a simple relationship since the availability of the model in electronic format had to be considered as well. The measures all exhibit a fairly high construct validity as evidenced by the high rank-correlation. In conclusion, the more sophisticated size metrics appear to be the most valid because they do not favour the shallow models above the more fully specified models.

Model ***correctness*** is measured against the formal modelling notation used by the model. Models could be rated using a composite "correctness score" measuring the number of errors, the degree of inconsistency as well as the use of standards.

Models have a definite and distinct structure, which is expressed by three different constructs: diagrams, groupers and inheritance tree.

Although quite a few ***grouping and diagram***-related metrics were investigated, overall, these metrics appeared to be not very useful for purely analytical comparative analysis purposes.

There is a plethora of metrics measuring the ***inheritance*** structure. The following were calculated for the models: number of "Is-A" relationships, use of multiple inheritance, number of unique entities in inheritance graph, inheritance graph size, inheritance connectivity density, inheritance depth / hierarchy nesting level, mean depth of inheritance tree, inheritance width, widest level, number of root classes, average number of children (NOC), standard deviation NOC, number of super-classes, reuse ratio and specialization ratio. The inheritance ratios give a fair feeling for the shape of the inheritance tree, and the degree to which inheritance is used in a model, but the interpretation of the metrics for comparing models leaves much to be desired. The validity of these metrics at this level of

analysis should be questioned, though there may well be a different case for their use at the design level.

The following model ***complexity*** metrics were calculated for the models: McGabe's cyclomatic complexity; absolute and relative connectivity; average, maximum and spread of fan-out; and DeMarco's data bang. Applying these classical software engineering metrics to the field of high-level (business) domain modelling was not very successful for the purposes of model analysis and their validity in the context of modelling appears suspect.

A ***graph plotting package***, such as Pajek, was very useful in visualizing the connectedness of a model, as long as entities were plotted in random order. When comparing models of similar size, plotting the models using the Fruchterman-Reingold minimum-energy algorithm allowed the visualization of the degree of relative spread of inter-relationship clusters, by looking at the amount of clustering of the lines or the amount of empty space in the graph. Other interpretations proved to be more problematic.

Combining the approaches from the graph analysis and system engineering metrics led to the idea of plotting and comparing ***the frequency distributions of the fan-outs*** for each model. The frequency distributions are indeed quite distinctive and characteristic of the underlying model. The "traditional" descriptive frequency distribution statistics, such as mean, dispersion, skewness and kurtosis, proved to be not very adequate for the type of distributions associated with fan-out frequencies due to extreme skewness and large number of outlier values. Consequently, ***alternative metrics*** were proposed: an intuitive categorization into two or three types of curves: waves, slides and roller-coasters; the harmonic mean; and overall bumpiness or smoothness (counting inflection points using a calculated threshold value to filter out noise).

As an attempt to measure ***architectural*** structure, the ***aesthetics*** of model diagram layout was analysed, by decomposing the layout aesthetics into 13 distinct attributes or dimensions. The degree of conformance of a diagram to each of the aesthetic dimensions can be calculated. These values can be combined to create a composite index. Many of the suggested aesthetic measures appear to provide a fairly valid approach to analysing the aesthetic dimensions of diagram layout. A suggestion was made as to how to weigh the different components in a way that the final overall composite aesthetic score is not subject to the particularities of any one formula.

Table 7-13 rephrases the summary conclusions more formally by means of explicit reference to the validity criteria as specified in 4.2.3. It must be noted that the specific validity issues are made in reference to the underlying model database and are *not* in any way indicative of the use of some of these metrics in the original context for which they were developed e.g. systems engineering and software development.

It is suggested that the metrics with a recommendation of "Y*" be subject to further external validity testing in future research projects.

Having completed the syntactic analysis, it is now possible to consider the second aspect of the model evaluation framework: the *semantic* model analysis.

Table 7-13: Summary of Syntactic Metric Validity Concerns.

Framework Criterion	Section	Metric / measurement	Recoommended?	Specific validity issues (refer 4.2.3) in respect of the model sample.
Size	7.1	Number of entities	N	low content and construct validity
		CASE count	Y	higest external validity
		Adjusted CASE count	Y	high criterion validity, unknown external validity
Correctness; error-free	7.2	Syntax error score	Y	relatively low internal & criterion validity
Integrity; consistency	7.2	Consistency score	Y	relatively low internal & criterion validity
		Standards level score	Y	
Modularity	7.3	Nr of groupers	Y	
		Nr of group levels	Y	
		Nr of diagrams	Y	
		Nr of entities/diagram	N	low criterion and construct validity
		Nr of relations/diagram	N	low criterion and construct validity
Structure; hierarchy	7.3	Use of multiple inheritance	Y	
		Inheritance connectivity density	N	no criterion and low construct validity
		Max. Inheritance depth	N	low content validity
		Mean Inheritance depth	Y	
		Inheritance width (max. / mean)	N	low criterion validity
		NOC (average / st.dev.)	N	low criterion validity
		Reuse ratio	?	some criterion and construct validity
		Specialisation ratio	N	low construct validity
Complexity; density	7.4	Cyclomatic complexity	N	not normalized, low criterion/construct validity
		Absolute connectivity	N	not normalized, low criterion/construct validity
		Relative connectivity	Y	fair construct & criterion but low content validity
		Average fan-out	Y	fair construct & criterion but low content validity
		Max. / st. dev fan-out	N	low criterion, content, construct validity
		De Marco's data bang	N	low face, criterion, content validity
		Relative data bang	N	low content validity
	7.5	Graph network plot: random	N	low criterion, content and construct validity
		Graph plot: Kamada-Kawai	N	low criterion, content and construct validity
		Graph plot: Fruchterman-Reingold	?	Only for similar-sized models, low criterion validity
	7.6	Fan-out: mean, mode, median, st. dev., skewness, curtosis	N	Low construct and criterion validity
		Fan-out: harmonic mean	Y	
		Fan-out distribution chart	Y	High face and content validity
		Fan-out model signature	Y*	Unknown criterion and external validity
Architectural style	10.1	Architecture temperature	N	Low construct and criterion validity
	7.7	Layout aesthetics	Y*	high internal and construct validity but questionable criterion, face & external validity

Chapter 8: Semantic Model Analysis

8.1 Introduction

This chapter deals with the second aspect of the framework: semantic analysis. The previous and next chapters deal with the other two aspects of the framework, namely the syntactic and pragmatic analysis respectively.

The semantic analysis of models is concerned with the intrinsic *meaning* of the model i.e. its relationship and mapping to the underlying domain reality it is representing. The syntactic analysis is content to deal with the structural relations i.e. shape and form of the entities and their relationships and groupers, and therefore treats all entity and relationship names as mere "alphanumeric labels". The semantic analysis looks at what the entities or relationships actually represent i.e. their correspondence to the underlying enterprise domain: is "the customer" modelled? Is a distinction made between corporate and private customer? Is there a distinction between lead, prospect, potential customer, actual customer, frequent customer, non-active customer? Is the entity representing "the customer" named client, customer, member, Client/Customer, Actual_Customer or CUST? Ideally semantic analysis also looks at the different attributes defining each entity (and the relationships in which it participates), the logic of the groupings, the correctness of, and amount of detail contained in, the relationship specifications.

The essence of semantic analysis is to unravel the meaning of the name (label, word, token) used for a specific model element (entity, relationship, grouper), i.e. semantic analysis is concerned with the correspondence (mapping, projection, validity) between the model (abstract or intellectual construct) and the underlying domain (reality).

Whereas syntactic analysis is fairly technical and easy to automate, semantic analysis involves the more difficult matter of meaning and interpretation and thus lends itself not quite as easily to objective and/or automated analysis. This is a fairly under-researched field in the modelling discipline and this section must therefore be seen as much more exploratory in nature. The emphasis is on a survey of possible "lines of attack", and on the critical evaluation of their relative merits and drawbacks rather than on strict prescriptions about which thesaurus or wordlist to use. Also, a conscious attempt is made to emphasize those analysis techniques which can in principle be automated, regardless of the domain being modelled. Because semantic analysis is by its very nature very dependent on the underlying (real world) domain being modelled, the focus is on principled approaches rather than on actual prescriptions or final recommendations. Also, because of the lack of research in this area, a number of "false starts" and techniques which yielded no successful or meaningful results will also be discussed, in order to guide future researchers who may want to build on the approaches suggested in this section.

Finally, it is important to note that in most ontological and certain linguistic research, similar analysis methods are used but these methods are presented in a much more formal language. Although a formal approach is important – especially to highlight the distinction between the terms used and the underlying concepts being denoted – the notation tends to obscure the actual analysis. Therefore a less formal approach in describing the various analysis methods has been adopted in this research.

8.2 Overview of proposed semantic analysis techniques

The proposed structured set of semantic analysis techniques is guided by the following list of questions which probe model semantics. The questions must also be seen in the light of the original meta-model employed to capture the enterprise models.

- How powerful is the model's meta-language or modelling technique in representing the richness of the underlying domain? If a relatively expressive modelling language is used, how well does the model make use of the rich constructs available in the modelling language?
- How much of the overall domain is captured? This relates both to scope and the level of detail:
  - The width of the model: Are all areas of the domain covered? Are all the essential concepts within the domain represented?
  - The depth of the model: To what level of granularity are concepts represented with emphasis on the finer distinctions between closely related concepts; sub-classes or specializations of concepts; how many of its attributes are captured and how well is the concept defined or described in the documentation?
- How well does the language used in the model map onto the vocabulary as employed by the users or readers of the model? This looks at the naming of model elements as well as accompanying model documentation (as provided by the model "producers") and compares it to the existence, comprehensibility or meaning differences of these terms in the vocabularies of the model "consumers".
- If several alternative models claim to cover the same domain, how much overlap (as measured in terms of domain coverage) exists between these models?

The above questions are perhaps not exhaustive but appear to cover the major semantic aspects mentioned in the different frameworks suggested earlier. These questions can be mapped to analysis techniques under the following headings:

- **Expressiveness**: what aspects of reality can be captured by the modelling language used for the model and how well does the model make use of the modelling constructs?
- **Completeness**: what proportion (depth and width) of the domain has been modelled?
- **Perspicuity** and **readability**: how understandable and self-describing is the model (to its intended audience)? This analysis can be applied to the model content itself as well as to its documentation. However, most authors consider the quality of documentation separately, so this will be discussed under a separate heading.
- **Relative correspondence**: how much overlap is there between the different models?

Note that expressiveness is determined by the modelling language adopted by the model and therefore measured differently than the other criteria. However, because this language represents a choice by the modeller and places constraints on the expressiveness of the model, it is an important semantic criterion for model evaluation. In addition, the modeller may decide to use all or only selected constructs provided by the modelling language, thereby further constraining the model expressiveness (as discussed in 8.4).

A final type of semantic analysis is applicable only in the light of the specific domain of the models in this research i.e. generic enterprise models:

- **Genericity**: how universally applicable is the model – can it be applied to any type of organization e.g. non-profit service organizations?

The discussion in this chapter follows the order of the criteria as listed in Table 5-8 with the exception of the criteria of "model similarity" (8.7) and "model completeness" (8.8). Notwithstanding the "second dimension" of the framework, these two criteria have been moved to the back of the chapter because they rely on techniques which are more naturally developed and explained in the context of the other semantic criteria.

8.3 Genericity

8.3.1 Description

Although the term genericity does not appear in the dictionary, it is a commonly used term in object-oriented software to denote "the capacity to produce components which have a general use. Such components are essential if we are to promote the reuse of code." [TYRR95] In the context of *generic enterprise models*, the genericity refers to the capability of the enterprise model to be applied to different organizations. The term genericity is preferred over the word "generality" because the latter is derived from "general" and has the connotation of lacking in precise detail, which is not the meaning intended: the enterprise models should be both generic (not general) and detailed.

The analysis suggested below is an attempt at a more structured approach to measuring genericity and should be seen as exploratory. An alternative approach – using a high-level framework – was also considered and is mentioned in Chapter 10.

Checking the appropriateness or applicability of a model to an arbitrary organization really has two aspects:

- How much of the organization (the domain) is not addressed by the model i.e. essential domain aspects that are not mapped by the model?
- How many of the model elements are not applicable to the organization (domain) under consideration?

This is illustrated in Figure 8-1.

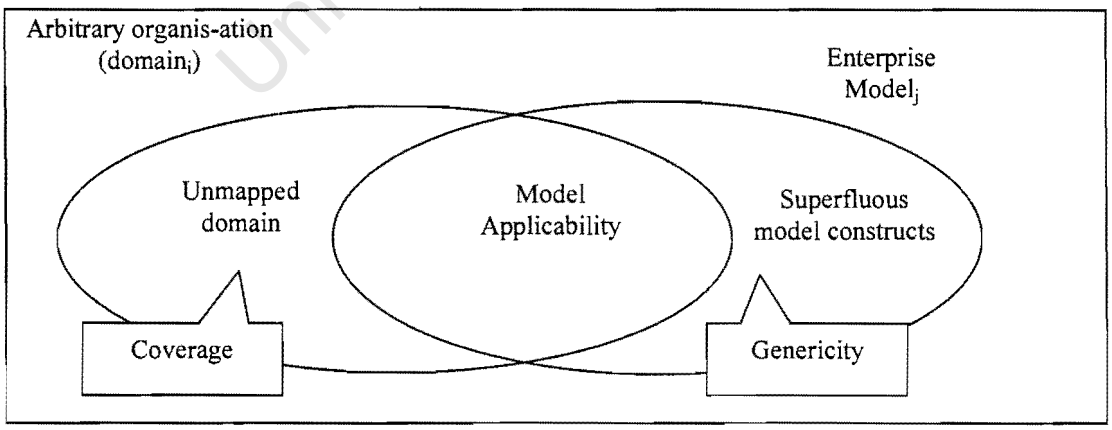


Figure 8-1: Model Coverage and Model Genericity.

Note the strong relation to model specificity (as measured by the model size): a high-level model cannot be expected to address a lot of the domain and a large, specific model is likely to include constructs which cannot be applied to all organizations. The degree to which the various enterprise

models map the domain is discussed below as a separate issue: domain *coverage*, but a provisional, subjective measurement will be made here.

8.3.2 Possible Measurement

What are the different types of organization encountered in the real world? Organization theory distinguishes different organization types based on structural aspects such as matrix versus hierarchical organization [DAWS96; EVAN93]. For purposes of this research, this is unlikely to affect many of the modelling constructs. Instead, a more pragmatic approach will be used, based on observations of the difficulty of porting enterprise information systems between organizations.

For discussion purposes, five different types or classes of organization will be distinguished, moving away from the prototypical manufacturing business. The following are the different types of organization against which each model will be ranked in applicability:

1. A medium-sized to large manufacturing organization, typically with at least one plant and a fairly elaborate but standard administrative support structure. This is the prototypical business with a hierarchical organizational structure.
2. A not-for-profit, large service organization such as a university or (non-trading) government agency. The lack of profit motive and physical product delivery differentiate it from the prototype.
3. A medium-size virtual organization with all physical infrastructure outsourced, such as a search engine company or e-book vendor. Typically there is no (or little) physical plant or physical location and a very flexible matrix type of organizational structure. (Note that the term virtual organization is also used in a different sense for temporary, project-based dynamically configured cooperative ventures. These are not considered here because, apart from the temporal aspect, they can easily be classified as any one of the other organizations listed here, depending on the nature and size of their business).
4. A owner-operated, non-franchise micro-business such as a small grocery, video-shop or bed & breakfast. It typically lacks a formal organizational structure (no functional division or management level) and procedure. Financial information requirements are assumed to be minimal.
5. A small non-government organisation (*NGO*) or community organization, including a sports club, a local charity or an arts association.

It must be noted that is not meant to be an exhaustive listing of all possible forms of organization or a theoretically sound organizational theory. It is a pragmatic gradual "expansion of the definition of organization".

To measure genericity, the applicability of each model will be rated against each type of organization. Table 8-1 lists the subjective scale ratings.

The assessment or rating of each model is admittedly a very subjective process, based on inspection of the model elements (especially the entities) of each model in the database. Methods were investigated to automate the mapping, e.g. by using sample "standard descriptions" of the organizations used in business analysis cases, but this proved to be a methodological minefield and was abandoned.

The method is therefore subject to bias and should be seen as experimental, with considerable scope for future methodological improvement. Consequently the ratings should not be seen as definitive, or

not even absolute, but are rated relative against the other models with possible room for interpretative differences. However, it is hopefully suggestive of a feasible approach.

Table 8-1: Model Genericity Rating Scale.

Rating	Description
1	Very low applicability; only a few model constructs can be mapped to the domain. It is probably better to build a model from scratch than to try to apply the model to the domain.
2	Slight applicability; a relatively large proportion of the domain is not modelled and much of the model is not applicable to the domain. Although an inspection of the model could be useful as background information, it is not to be used as a guideline for modelling the domain.
3	Fair applicability; many concepts in the domain are mapped, although a reasonable number may need to be re-labelled. With a certain amount of work, the model could be adopted/extended for the domain although the model is a useful start for the domain analysis. Many model elements may be redundant or unnecessary for modelling a particular type of organization.
4	Good applicability; a large area of the domain is mapped and a good number of the model elements correspond directly, without re-naming, to the domain although there may be a fair number of superfluous model constructs. The model could well be adapted to suit the domain without the need for a fresh domain analysis.
5	Excellent applicability. Virtually all of the model constructs apply to the domain. Some of the finer details of the domain may need to be added, but most of the model is directly applicable. Note: no model covers the entire domain or has no redundant element.

8.3.3 Application and Analysis

Table 8-2 lists the ratings for both aspects: how well each model covers the domain and how much of the model is superfluous (redundant) when considering the various types of domains.

Table 8-2: Domain Coverage and Model Genericity Ratings and Ranking.

	Domain Coverage							Model Genericity								
Model	Manufacturing	Service Org	Virtual Org	Micro-business	Community/NGO	*Average*	*Ranking*	Manufacturing	Service Org	Virtual Org	Micro-business	Social Org	*Average (G1)*	*Ranking (G1)*	*Average StDev (G2)*	*Ranking (G2)*
AIAI	4	4	3	3	3	***3.4***	***10***	5	5	5	5	4	***4.8***	***2***	***0.50***	***4***
AKMA	4	5	4	4	3	***4.0***	***3***	5	5	5	4	3	***4.4***	***5***	***0.80***	***7***
ARRI	4	3	4	4	2	***3.4***	***10***	5	4	4	5	3	***4.2***	***7***	***0.87***	***10***
Baan	5	4	4	4	3	***4.0***	***3***	5	3	3	2	1	***2.8***	***18***	***1.10***	***19***
BelgAcc	4	4	3	4	2	***3.4***	***10***	5	5	5	4	3	***4.4***	***5***	***0.89***	***14***
BOMA	3	4	4	3	2	***3.2***	***13***	5	5	4	4	3	***4.2***	***7***	***0.84***	***8***
CYC	5	5	5	5	5	***5.0***	***1***	3	3	3	1	2	***2.4***	***21***	***0.45***	***3***
Fowler	3	3	3	3	2	***2.8***	***16***	5	5	5	4	4	***4.6***	***3***	***0.50***	***4***
Hay	4	4	4	4	2	***3.6***	***9***	5	5	4	2	1	***3.4***	***12***	***1.36***	***21***
Inmon	4	5	4	4	3	***4.0***	***3***	5	5	4	4	3	***4.2***	***7***	***0.77***	***6***
Miller	2	2	2	2	2	***2.0***	***22***	5	5	5	5	5	***5.0***	***1***	***0.00***	***1***
NHS	3	4	4	3	2	***3.2***	***13***	3	3	3	2	1	***2.4***	***21***	***0.87***	***10***
Nippon	5	4	3	3	1	***3.2***	***13***	5	3	3	1	1	***2.6***	***20***	***1.58***	***23***
Ott.-Big	3	3	3	3	2	***2.8***	***16***	4	4	4	2	1	***3.0***	***17***	***0.93***	***15***
Ott.-Dense	3	3	3	3	2	***2.8***	***16***	4	5	4	2	1	***3.2***	***14***	***1.05***	***17***
Purdue	5	5	4	4	2	***4.0***	***3***	5	4	4	2	1	***3.2***	***14***	***1.43***	***22***
Random	1	1	1	1	1	***1.0***	***23***	1	1	1	1	1	***1.0***	***23***	***0.00***	***1***
SAP	5	4	4	4	3	***4.0***	***3***	5	3	3	2	1	***2.8***	***18***	***1.10***	***19***
Semi-Rand	3	3	3	3	2	***2.8***	***16***	4	5	4	2	1	***3.2***	***14***	***1.05***	***17***
SanFran	3	3	3	2	2	***2.6***	***20***	4	5	5	4	2	***4.0***	***10***	***0.89***	***12***
Silverston	5	4	4	4	3	***4.0***	***3***	5	4	4	2	2	***3.4***	***12***	***1.02***	***16***
TOVE	5	5	4	4	3	***4.2***	***2***	5	4	4	3	3	***3.8***	***11***	***0.84***	***8***
USB	3	3	2	3	1	***2.4***	***21***	5	5	5	5	3	***4.6***	***3***	***0.89***	***13***

The first set of data columns gives a rough indication of how much of the domain is covered for different types of organizations. The second set is at first sight the more important data set: how many of the constructs apply to each type of organization.

As a first suggested measure of genericity G1, the (unweighted) average score for each model serves as a quantitative expression of how well all model elements can be applied to different types of organizations.

When ranking the scores, the most generic model is Miller's Living Systems model, not surprising since it was designed to apply to any living system ranging from a simple biological cell to supra-national institutions. Following a close second is AIAI's fairly abstract and high-level enterprise ontology, followed by the USB generic financial forecasting model and Fowler's high-level patterns. In general, the G1 score seems to be a fairly valid measure, although it appears that larger models are penalized due to the large number of their constructs that are not applicable in many situations.

In particular, there is a validity concern with the ranking of the CYC model, which has many highly generic concepts but also a lot more specific constructs. This suggests a possible alternative genericity measure,G2, which looks not at the actual scores but how the scores change when the domain (the type of model) is changed. Note that this is a significantly different, but possibly equally plausible interpretation of the concept of genericity. If a model's rating of coverage stays constant, regardless of which domain is being modelled, and the number of redundant concepts also stays fairly constant, then it can be argued that the model has many generic constructs across the different domains. Thus one can operationalize G2 as the average of the standard deviations of the scores for both the domain coverage and the model genericity ratings. These are given in the second last column and ranked in the last column.

For many models, G1 and G2 appear to correlate quite closely, but there are a few very significant shifts:

- The CYC model makes a huge move forward, because of its constantly high ratings for covering any of the domains. This is in line with the different interpretation of genericity.
- Initially surprisingly, the Random model, which previously was ranked last, now shares first position. This can be explained on technical grounds: virtually none of its constructs is applicable to the domain, hence G1 should be minimal. However, it is very consistent across the different domains i.e. it models all different organizations equally (perfectly) badly, hence the high G2. This confirms, in a way, the validity of both measurements, but also the need to consider both measures separately without combining or averaging them.

For all ratings, the "manufacturing" oriented models fare comparatively and perhaps unfairly badly. The ERP models SAP and Baan, as well as the CIM models Purdue and Nippon, all score relatively low. This is perhaps somewhat unfair, due to our operationalization of different "types" of organizations. Their authors probably never intended them to be applied to a small business or NGO, but rather across different manufacturing industries (with perhaps a bias towards those with continuous and mass production processes). It is, however, perfectly possible to retain the definitions for G1 and G2, but to change the original table to use organizations from different industries, instead of the five categories mentioned above. The method remains the same, but the interpretation of the "range" of genericity is different.

8.3.4 Conclusion

In conclusion, it must be admitted that the methodology for producing ratings is subjective, and that the interpretation of the meaning (or scope) of genericity is subject to debate. In particular the following different interpretations were uncovered:

- Does genericity apply across different industries or across different types of organizations?
- Should genericity be measured by what fraction of the model constructs is applicable on average to all domains, or by how consistently the high-level constructs apply across the range of different organizations?

Despite these concerns, however, the method suggested above appears to have sufficient face validity to warrant its use as a rough metric. In addition, it is amenable to modifications to suit the different possible situations and interpretations. It appears that the genericity measures are perhaps the most problematic metrics suggested in this research, though luckily the least critical: when applying the framework to other models (i.e. non-generic models), this measure is not relevant.

8.4 Expressiveness

8.4.1 Description

The expressiveness of a model refers to how much information a model has captured about the various model elements. It refers to the richness of the modelling language used. For example, the definition of a relationship *has* to contain a reference for the "To-Entity" and the "From-Entity", but some models will also provide a definition and a name for the relationship. Other models go even further and include cardinalities and role names for bi-directional relationships. Very few models provide full definitions or descriptions for the models as part of the documentation. Other models define different types of relationships and some specify constraints over a number of relationships.

The expressiveness is not necessarily related to the amount of modelling information that is actually provided. The extent of the model can be measured syntactically by its size, or semantically by the domain coverage (or completeness) and can be likened to a quantity measure: it measures the (semantic) "width" of a model, the number of model elements. Expressiveness is more qualitative because it measures the amount of detail i.e. the semantic "depth" of the model: the number of attributes for each model element. An analogy that could be used here is the comparison of a well-spoken person who has nothing to say as opposed to a person with a lesser command of the language but much more extensive knowledge or experience – the ideal situation is to combine both: a well-spoken (model expressiveness) and knowledgeable (domain coverage or model completeness) person.

The expressiveness of a model is intrinsically tied to the modelling language used. The UML class diagram is a richer modelling language than the classic ERD, since the former allows for various types of relationships including generalization, aggregation and composition as well as constraints over relationships, but UML is not as rich as KIF which allows you to define these relationships more formally, including e.g. default values, domain ranges etc. A more detailed discussion of modelling languages and meta-modelling concepts is found in Chapter 3.

However, not all models expressed in a particular language necessarily make use of the power of the language: merely re-drawing an ERD using UML class-diagram symbols does not make it richer or more expressive. Hence measuring the expressiveness of the various enterprise models looks at the modelling language constructs as well as at the extent to which the models implement the various features of these modelling languages.

8.4.2 Measurement

An approach to measuring the expressiveness of modelling notations was suggested in [MCLE98]. In attempting to quantify the "complexity" of a methodology, he used a composite weighting index, analogous to the function point metric, to rate how much of the semantics of a modelling notation was used in a given methodology.

In general, the expressiveness of the enterprise model will be measured against the meta-model attributes used for capturing the data as discussed in Chapter 4. Some exceptions were made where the attribute does not really measure model expressiveness e.g. the use of internal codes for model elements does not enhance the expressiveness. The following diagram gives an overview of the important contributors to a model's expressiveness, with the last column reserved to identify those models that formally added some model information beyond what was required in our meta-model – for the most part this referred to relationship constraints (e.g. UML allows for an OR or AND bracket across different relationships linking a given concept e.g. a given order can be placed by an organization OR an individual but not both at the same time). Table 8-3 is a summary of some of the information given in Appendix B.

Table 8-3: Scores for Model Expressiveness Attributes.

Model ID	Main modelling language	Degree of Formality	Diagrams?	Directed graph?	Nr. of Generalizations	Depth of inheritance tree	# Multiple Inheritance rels	Nr Groupers (level 1+2)	Nr Grouper Levels	Entity Definitions?	Entity Examples?	Entity Attributes?	Relationship Names?	Relationship Role Names?	Relationship Cardinalities?	Relationship Types?	Rel./Group Definitions?	Additional e.g. constraints
AIAI	KIF	High	N	N	98	6	11	91	2	Y	N	S	Y	N	Y	S	Y	Y
AKMA	EERD	Med	Y	N	33	2	0	6	2	Y	Y	Y	Y	N	Y	N	N	Y
ARRI	IDEF0	Med	Y	Y	70	3	0	6	2	Y	N	N	Y	N	N	Y	N	N
Baan	RDBMS	Med	N	Y	142	4	0	8	3	Y	N	Y	N	N	N	N	N	N
BelgAcc	Dutch	Low	N	Y	462	4	0	13	4	N	N	N	N	N	N	N	N	N
BOMA	UML	M-Hi	Y	S	115	5	0	42	3	Y	S	Y	S	N	N	Y	N	S
CYC	CycL	High	N	Y	654	10	182	43	2	Y	N	N	Y	N	N	Y	Y	N
Fowler	Odell	Med	Y	N	69	4	0	55	3	N	N	N	S	N	Y	N	N	S
Hay	EERD	Med	Y	N	180	5	19	91	3	N	N	Y	N	Y	Y	N	N	Y
Inmon	"ERD"	Low	Y	Y	220	3	0	73	3	N	N	Y	N	N	Y	N	N	N
Miller	English	Low	Y	Y	44	5	0	2	1	Y	Y	N	N	N	N	N	N	N
NHS	UML	Med	Y	Y	236	6	0	1	1	Y	N	Y	Y	N	Y	N	N	N
Nippon	"DFD"	Low	Y	Y	146	5	0	16	5	Y	N	N	Y	N	N	S	N	N
Ott-Big	English	Low	N	Y	0	0	0	1	1	Y	N	N	N	N	N	N	N	N
Ott-Dense	English	Low	N	Y	0	0	0	1	1	Y	N	N	N	N	N	N	N	N
Purdue	"DFD"	Low	Y	Y	103	3	0	12	3	Y	N	N	Y	N	N	N	N	N
Random	English	Low	N	Y	320	15	0	1	1	N	N	N	N	N	N	N	N	N
SanFran	UML	M-Hi	Y	N	11	3	0	40	2	N	Y	Y	Y	N	N	Y	N	N
SAP	EERD	Med	Y	N	160	4	34	17	3	N	N	Y	S	N	Y	Y	N	N
Semi-Rand	NA	Low	N	Y	320	15	0	1	1	N	N	N	N	N	N	N	N	N
Silverston	EERD	Med	Y	N	82	4	0	62	3	N	N	Y	N	Y	Y	S	N	Y
TOVE	KIF	High	N	N	72	4	5	85	4	Y	N	S	Y	N	Y	N	N	Y
USB	Excel	High	S	Y	258	2	0	19	4	N	N	N	N	N	N	Y	Y	Y

Note that the following abbreviations were used:

- Y = Yes (=all or most); N = No (= none or few); S = Some;

- Med = Medium; M-Hi = Medium-to-High.

In order to provide a single, overall expressiveness measure, it is suggested that the various components are combined by means of calculating a weighted average. Each of the relevant attributes (or components) of expressiveness is converted into a value by means of the following mappings:

- The categories from Low to High were awarded values as follows:

Low	Medium	Medium-High	High
0	1/3	2/3	1

- The categories Yes/No were awarded values as follows:

No	Some	Yes
0	½	1

- Generalization: "Yes" if used at all i.e. "# Genls" > 0.
- Depth of inheritance tree (note that the model already gets a score for having generalization i.e. in that case the depth of the inheritance tree must be at least 2):

<3	3	>3
0	½	1

- Multiple inheritance: "Yes" if used at all i.e. > 0.
- Number of grouper levels (note: if = 1 then no real grouping is used)

<2	2	>2
0	½	1

A second decision is to decide on the appropriate weights for each measure. Suggested weights are given in the row labelled "MaxScore" in the table and motivated as follows:

- The degree of formality of the modelling language is very important, because it is a very good measure of the expressiveness of the model. Using a formal language requires the modeller to be very explicit about any of the assumptions within the model. It was therefore given a weight of 3.
- Although a diagram is not necessarily more expressive, it eases model inspection and adds clarity. It was therefore included though an argument against this stand could be considered equally valid, depending on the intended model use.
- A directed graph required additional information about which way the relationship is travelled.
- Generalization relationships provide important semantic information. If several levels are used, the model becomes significantly richer and more complex. Additional complexity is introduced by means of multiple inheritance. Overall, a maximum score of "3" is introduced by full use of inheritance. It must be noted that the explicit use of a generalization is required in order to score, e.g. the "natural language" models include some generalization concepts but do not define this relationship explicitly, so they do not score in this area.
- Although entity names are sufficient for the simplest model, it is desirable to have examples, definitions (or descriptions) and attributes for entities. An overall score of "3" can be added if a model includes all three entity attributes.
- Relationships can have much richer information. Few models provide names for relationships, although the more advanced provide role names for both the entities participating in the (binary) relationship. In addition, entity cardinalities provide additional information. (Note that where attributes are included for relationships, e.g. in the SAP model, these relationships were captured

as entities). Generally a model provides either relationship names or role names, not both. Role names therefore carry double the weight of "single" relationship names, since they express twice as much information. Some models explicitly express different relationship types (as allowed in UML, for example), which require additional conceptual model analysis. Using relationship types therefore contributes to an increased expressiveness score.

- Finally, some models include information that was not captured or included in our meta-model. Most of this information related to constraints, e.g. over relationships or attribute domains.

Table 8-4 calculates the proposed overall weighted expressiveness score.

Table 8-4: Overall Weighted Model Expressiveness Score.

Model ID	Degree of Formality	Diagrams?	Directed graph?	Use of Generalization	Depth of Inheritance Tree	Multiple Inheritance	Nr Grouper Levels	Entity Definitions?	Entity Examples?	Entity Attributes?	Relationship Names?	Relationship Role Names?	Relationship Cardinalities?	Relationship Types?	Rel./Group Definitions?	Additional e.g. constraints	*Expressivenss Score*
MaxScore	***3***	***1***	***1***	***1***	***1***	***1***	***2***	***1***	***1***	***1***	***1***	***2***	***1***	***1***	***1***	***1***	***20***
AIAI	3	0	0	1	1	1	1	1	0	0.5	1	0	1	0.5	1	1	**13**
TOVE	3	0	0	1	1	1	2	1	0	0.5	1	0	1	0	0	1	**12.5**
BOMA	2	1	0.5	1	1	0	2	1	0.5	1	0.5	0	0	1	0	1	**12**
CYC	3	0	1	1	1	1	1	1	0	0	1	0	0	1	1	0	**12**
Hay	1	1	0	1	1	1	2	0	0	1	0	2	1	0	0	1	**12**
Silverston	1	1	0	1	1	0	2	0	0	1	0	2	1	0.5	0	1	**11.5**
SAP	1	1	0	1	1	1	2	0	0	1	0.5	0	1	1	0	0	**10.5**
USB	3	0.5	1	1	0	0	2	0	0	0	0	0	0	1	1	1	**10.5**
AKMA	1	1	0	1	0	0	1	1	1	1	1	0	1	0	0	1	**10**
SanFran	2	1	0	1	0.5	0	1	0	1	1	1	0	0	1	0	0	**9.5**
NHS	1	1	1	1	1	0	0	1	0	1	1	0	1	0	0	0	**9**
ARRI	1	1	1	1	0.5	0	1	1	0	0	1	0	0	1	0	0	**8.5**
Nippon	0	1	1	1	1	0	2	1	0	0	1	0	0	0.5	0	0	**8.5**
Fowler	1	1	0	1	1	0	2	0	0	0	0.5	0	1	0.5	0	1	**8.5**
Baan	1	0	1	1	1	0	2	1	0	1	0	0	0	0	0	0	**8**
Inmon	0	1	1	1	0.5	0	2	0	0	1	0	0	1	0	0	0	**7.5**
Purdue	0	1	1	1	0.5	0	2	1	0	0	1	0	0	0	0	0	**7.5**
Miller	0	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	**6**
BelgAcc	0	0	1	1	1	0	2	0	0	0	0	0	0	0	0	0	**5**
Random	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	**3**
Semi-Rand	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	**3**
Ott-Big	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	**2**
Ott-Dense	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	**2**

8.4.3 Interpretation

The following observations can be made:

- The most expressive models are the ontology-based models AIAI, TOVE and CYC, along with some of the data modelling such as BOMA (which includes JAVA source), Hay and Silverston.
- As an example of how the deficiencies of a modelling language can be overcome, SAP makes full use of all the EERD modelling capabilities to produce very rich (expressive) diagrams whereas

models such as SanFran or Fowler do not make use all the capabilities of the more expressive UML language and actually score lower.

- Least expressive are the "minimalist" Ottawa and Random models, which were specifically included because of, or designed for, minimal syntax and semantics.
- Almost equally low in expressiveness are the "natural language" models Miller and BelgAcc; their scores are almost low by necessity due to the lack of formality of the language. It must be emphasized here that this is very much a consequence of the way in which the various expressiveness attributes are operationalized. A contrary argument could be that natural language is much more expressive than formal languages, but it is here considered that richness without precision should not be considered as expressiveness in the realm of enterprise modelling.
- Finally, there is a group of models that use a fairly expressive modelling language but fail to employ its expressiveness: ARRI, Fowler, Nippon, Baan, Inmon, and Purdue all fail to make full use of the modelling constructs available to them. These models are less expressive, and therefore less preferable from an expressiveness perspective, than some of the other models within the same reference discipline and using the same modelling language. These models could easily increase their expressive quality by attending to the deficient areas as identified by 0 or ½ scores for the various expressiveness attributes, although this requires additional business analysis input. With respect to the Baan model, it must again be emphasized that the scores are for the model that was reverse engineered from the relational data tables; there is no reason to believe that the *original* Baan model is less expressive than its SAP counterpart.

8.4.4 Validity

Overall, the quantification of an expressiveness score (or index) appears to be a feasible and valid approach. The key to the construction of the expressiveness score is the availability of a suitable and sufficiently detailed meta-model. This model can then be used to map the use of the various modelling language constructs to the attributes of the model elements in the meta-model.

A fairly simplistic scoring and weighting mechanism can be used to obtain a combined expressiveness score. The final resolution of expressiveness score is obviously dependent on the exact weights allocated to the various expressiveness attributes as well as on the way in which they are measured, and a difference of ½ or 1 in the score should not be interpreted as a significant difference in expressiveness. However, it appears that the overall score is a good and valid indicator of how expressive a model is.

As an interesting validity test, the expressiveness score was recalculated without weights (or all weights set to 1). Table 8-5 lists the results.

It is clear from Table 8-5 that, for the majority of the models, no major change in position occurs. The unweighted ranking for most models is to within one or two positions of their ranking using the weighted expressiveness score, which makes it a fairly robust measure and increases the confidence in its validity.

8.5 Model Perspicuity and Readability

The concepts of model perspicuity and readability refer to the extent to which the model can be understood or comprehended by the intended users or readers of the model and how self-describing the model is. This is not an absolute measure because it is dependent on the language used by the

model users. Further complicating this issue is the fact that models often have different groups of model users, each using their own jargon, e.g. IT professionals versus business managers.

Since many of the models in the sample have been created by computer scientists or IT professionals, a particular concern is the perspicuity of the model terminology – especially from the perspective of people working within the business domain. The term model perspicuity is used here in preference to model comprehensibility or understandability because the latter terms often imply a dependence on structural model characteristics such as model complexity which is a syntactic measure.

Table 8-5: Effect of Weighting on Model Expressiveness Score.

Model ID	*Weighted Expressivenss Score*	Model Rank - Weighted ES	Unweighted Expressiven Score	Model Rank - Unweighted ES
AIAI	13	1	10 1/2	1
TOVE	12 1/2	2	9 1/2	3
BOMA	12	3	9 2/3	2
CYC	12	3	9 1/2	3
Hay	12	3	9 1/3	5
Silverston	11 1/2	6	8 5/6	6
SAP	10 1/2	7	8 5/6	6
USB	10 1/2	7	7 1/2	11
AKMA	10	9	8 5/6	6
SanFran	9 1/2	10	7 2/3	10
NHS	9	11	8 1/3	9
ARRI	8 1/2	12	7 1/3	13
Fowler	8 1/2	12	6 5/6	14
Nippon	8 1/2	12	7 1/2	11
Baan	8	15	6 1/3	18
Inmon	7 1/2	16	6 1/2	15
Purdue	7 1/2	16	6 1/2	15
Miller	6 1/2	18	6 1/2	15
BelgAcc	5	19	4	19
Random	3	20	3	20
Semi-Rand	3	20	3	20
Ott-Big	2	22	2	22
Ott-Dense	2	22	2	22

Thus, perspicuity is here defined as referring only to the clarity of the language or vocabulary used within the model. As mentioned above, this is dependent on the language capabilities of the audience (model consumer) and therefore a subjective or relative measure.

8.5.1 Proposed Analysis Methodology.

Conceptually, the proposed (generic) methodology for model perspicuity analysis is straightforward.

1. Make a list ML (Model Lexicon) of all the names of model elements. If several models are to be analysed, one ML_i for each model i must be constructed. This is likely to involve some conversions to make all lists conform to a common standard.
2. Make a list UL (User Lexicon) of the common domain vocabulary used by the model users. If possible, include a measure of frequency for, or familiarity with, each word.
3. Map (each) ML_i onto UL i.e. check how much of the ML is contained within UL and, conversely, how much of ML is not part of UL.

4. Calculate some overall perspicuity measure to reflect the proportion of ML contained within UL. Where possible, use a measure that can take into account the relative word frequencies if they are available.
5. The measure can then be interpreted against an absolute standard or, when comparing different models of the same domain, against the calculated perspicuity measures of the other models e.g. through ranking the different models. Alternatively, the perspicuity (rating) of a specific model can be improved by analyzing the concepts that could not mapped onto UL, and by checking whether they can be replaced by an equivalent synonym which is part of UL.

In practice, the methodology becomes quite a bit more complicated, not in the least due to the fact that several alternatives for the "domain vocabulary" exist. The following describes how the proposed methodology was operationalized in this research. Again, the emphasis is on a pragmatic and exploratory approach, along with suggestions for possible alternative approaches.

A very important *caveat* is in order here. The proposed methodology relies entirely on the matching of word "tokens", without taking into account the meaning. The meaning or intention of any given word (e.g. "order") may be very different to the meaning ascribed to that word by the user. This is a problem that cannot be resolved by using a lexicon-based approach. Unfortunately, the proposed methodology appears to be the most practical approach and is therefore a pragmatic approach to what might otherwise become an intractable issue. As a partial defence, it should be remembered that this caveat applies equally to all models and thus a consistent error or bias can be expected. It is important that this limitation to the semantic analysis be borne in mind for much of the remainder of the discussion in this chapter.

8.5.2 Step 1: Creating the model word lists ML_i

The first step was to prepare a list of words from the entity names for each of the models. Because of the wide variety in naming conventions, some pre-processing was necessary to ensure that all lists were comparable. Additionally, some decisions had to be made about which model element names to include in the lists.

Firstly, all documentation, model definitions and descriptions were excluded from the model lists. Although these are important, a separate analysis of model documentation will be done in the next section. The focus of this section remains purely with the names or labels of the model elements. Obviously, subsequent researchers may decide to make a different judgment call, and to include all terminology used in the model; this should not affect the principles of the analysis as detailed below. The main motivation for restricting the model perspicuity analysis to only the model element names, is the fact that this is the only information available for some of the models in the sample. Another reason is the fact that a large number of model users may not consult any of the documentation or detailed model element descriptions.

In principle, the analysis should be applied to all model entities used in the model. In this research, names of grouper entities have been excluded due to their small number and technical nature (many include chart numbers and abbreviations). A more critical matter is that of the relationship names. Not all models have named their relationships and many of the remaining models – those that have named the relationships – differ substantially on the way in which they apply the naming: some name only one direction, others apply names to both entity roles and in many cases models label only selected relationships. A further complication with relationship names is that many models use a few different names for many different relationships (e.g. "is used by", "participates in", "is a resource for",

"controls", "includes", etc.). Because this research focuses on the conceptual validation of various model analysis techniques, relationship names have not been included in the perspicuity analysis, as it is feared that this would introduce a substantial noise factor. Should the analysis be intended for real-world analysis of competing models, it should prove easy to include relationship as well as grouper names.

1. Because different models have different naming conventions, the following text processing was applied in order to standardize the model entity word lists.
2. Removal of all punctuation symbols and other non-alpha characters: _ - * / ^ ( ) [ ] 's ? , ; " ! & . These were replaced with a <space> to avoid concatenation of words such as "corporate/private customer" into "corporateprivate customer".
3. Removal of all digits.
4. Conversion of all UPPER-CASE to lower-case characters.
5. Removal of "noise words" (number of word removed): a (5), about (0), an (7), and (88), at (5), by (38), for (47), has (12), in (84), near (1), of (194), or (52), per (14), to (16), when (2), with (5).

The final step is probably the most contentious one. Since the most important and best validated "user vocabulary list" consists of single words (e.g. "income" separately from "gross"; as opposed to regular business vocabulary dictionaries which would have separate entries for "gross income", "net income" etc.), the decision was taken (very reluctantly) to split all multiple word entity names into separate word entries e.g. AIE016 "critical success factor" would be broken up into the three separate words critical, success and factor. A separate record was made of the fact that each of the constituent words should carry only a proportional weight of an original entity name; e.g. for the AIE016 example, the "critical" would carry one-third of the weight of a single word entity name. (Subsequent sensitivity analysis revealed that adjusting for weight does not affect the analysis significantly.) Although this procedure violates the meaning of the original term, it was done for pragmatic considerations. It is not expected that this biased the results in a significant way because, for perspicuity analysis, the correspondence of the meaning of a phrase noun such as "good long drink" against "agreeable elongated beverage" is less important that the actual readability of the constituent words. Additional support for this argument is the fact that readability statistics generally also do not take into account double words but checks word readability on the basis of the single words (as well as sentences and paragraphs).

To facilitate calculations, very long entity names were truncated after the first 8 separate words. In practice, virtually all of the very long entity names belonged to the BelgAcc model, with the longest entity being the 18-word "bezoldigingen premies pensioenen voor bestuurders zaakvoerders en werkende vennoten die niet worden toegekend uit hoofde van een arbeidsovereenkomst" [BEE272] – referring to executive remuneration outside a formal labour contract. Finally, it must be noted that concatenated entity names were not separated unless the constituent words were clearly indicated e.g. by hyphenation, underscores and other symbols, or capitalization. For example, CYC contained 582 concatenated terms such as "VisualInformationSource".

The result of step 1 is the creation of 23 MLs – lists of single words derived from the respective model's entity names – one ML for each model. Although the order within each ML is not important, it is useful to arrange the words in alphabetical order to facilitate processing. The above results in the creation of the following set of MLs:

$$\{ML_i \mid i \in \{AI; AK; AR; BA; BE; \ldots SR ; TO; US\}\}$$

with two-letter model acronym subscripts e.g. ML_{AI} denoting the word list for the AIAI model (see Appendix F).

8.5.3 Step 2: Creating the appropriate domain vocabulary word list.

The creation of a word list reflecting the use of domain terminology by model users is probably the most formidable obstacle from a methodological point of view. The following are some of the problems with creating a single list:

- One assumption is that the language of the model is the same as the language of the model user. The "BelgAcc" model is a significant illustration that this is not always true and therefore serves as a test case. All other models in the sample use English. Where an English model is to be employed by non-English speaking users, or vice versa, the analysis should still be valid: strict adherence to the procedure will indeed produce very low "perspicuity" ratings.
- Each geographical region, and even business, uses different subsets of the English language.
- Users may come from different disciplines, each employing their own "jargon".
- Even amongst users from the same background, individual vocabularies are likely to show a significant amount of variation.

Despite these methodological problems, some compromises are possible. In principle, the word list could be compiled from a large corpus of communications by the model (target) users. This is most easily achievable by compiling electronic communications, although this corpus will show a definite bias depending on the type of communications from which it is drawn: e-mail, correspondence, formal reports or studies, and verbal communications processed by speech recognition, are likely to produce different vocabulary lists.

In the realm of enterprise models, the prime candidate for the source of the vocabulary is the *business community,* since the model users (apart from IT professionals) are business people. After researching the various available linguistic corpora, the following two approaches were adopted.

1. A linguistics research project in the appropriate vocabulary to teach foreigners "English for Business Purposes" has produced a tagged "Business Letter Corpus" (*BLC*) which was based on a large variety of English business letters [SOME99]. The tagging includes absolute and relative word frequencies, a word level index (referring to the frequency or easiness of the word in common i.e. non-business specific English, but from the perspective of a non-native English speaker), word type (noun, verb, adjective etc.) and the identification of stem and derivatives e.g. the verb "control" being the stem or keyword for the "lemma group" which also includes "controls", "controlled" and "controlling" (each of the words being different "tokens"). [SOME99, p.56] defines a lemma as "[a] set of graphic words having the same stem and/or meaning and belonging to the same grammatical word class, differing only in inflection and/or spelling. Thus, go, goes, going, went and gone form a lemma or a lemma group." Only the base portion of the BLC was used, i.e. that portion consisting of 6408 entry words (or lemma groups) namely those lemmas which appeared at least five times in the total business languages source corpus consisting of 1,119,578 words. A second portion of the BLC contains the 13033 low-frequency words which are not considered representative of "business language", as exemplified by its first word (# 6409) "a-pr" and its last word (# 19441) "zyx"; although most of the other words in the list are regular but far less common English words.

- Where both words existed separately in the index the word was deleted e.g. "ground-level" was removed since both "ground" and "level" were existing lemmas in the BLC. 112 words were deleted.
- Where one or both word parts did not exist already as separate lemmas in the BLC, they were added (with the same frequency statistics) e.g. hi & tech (both terms added since neither existed separately in the BLC) or thought (from thought-provoking; provoke was an existing lemma in the BLC). This resulted in the addition of only 11 words.

7. 72 words had no "WL" rating. These are words that cannot be found in the reference corpora of common (i.e. non-business) English words and their common usage frequency could therefore not be determined. Most of these 72 were found to be misspellings or nonsensical words and were therefore deleted e.g. delp (help?); dolf (golf?); durin (during?); en (an?); fandangle; ferger (merger?); etc. Only 5 of the 72 words were found to be common but business-specific abbreviations and were therefore kept in the word list: JIT, RFP, VLSI, FAA, ABC. They were given a WL rating of 30, which is the lowest BLC rating possible i.e. the BLC words least frequently used in common English language.
8. At this stage 5202 lemma key words remained from the 6408 in the original BLC with an additional 4619 word form equivalents.
9. Another small set of word form equivalents was added after analysis of the electronic dictionaries, bringing the total number of words in the BLC word list to 9863.
10. All the original lemma entry words were re-ranked and words with equal frequencies were given equal rankings (the original BLC ranked words continuously in alphabetical order, even for words with the same frequency). The word form equivalents were given the same ranking, frequency and WL rating of their key word.

The dictionary entries were processed in a very similar manner. The following gives a brief overview of some of the more important steps.

1. Replacement of all non-alpha characters ( ) - / etc with spaces
2. Parsing of all multi-word entries into their constituent component words and creating separate entries for these in the word lists. This increased the total word list size from 9218 to 18007.
3. Removal of some nonsensical entries as well as those containing numbers.
4. Removal of all duplicate words which were mainly created during the parsing process. After duplicate word removal, the combined size of the 4 lists decreased to 7001 entries.
5. Since the dictionaries did not have lemma word groups, a different method had to be found to check for all different word forms of the same key word (control – controls – controlled etc.). Rather than create huge word lists containing all the possible word forms for all the dictionary words, an alternative approach was followed to minimize the size of the resultant word lists. A morphological analysis was done of the lists containing the *model* words (i.e. the MLs). These were checked using the most common English morphology algorithms for words who were present in plural, possessive, conjugated verb forms etc. (e.g. word endings in -ed, -s, -ing, and -tion). Where alternative word forms were found, these (and only these) alternative word forms added to the respective dictionary word lists where the base (key) word was present. For instance, if a model (word list) contained the word "controlling", the word "controlling" would be added to all the dictionary lists that already contained the entry "control" (or one of its other derivatives).

This resulted in a much smaller increase in dictionary word list size that would otherwise have been necessary. All in all, 987 words were added to the various dictionary lists.

6. Interestingly, this analysis revealed a number of word form alternatives that were missed by the researcher who compiled the BLC including e.g. power - powered; elect - electing; etc. These were added to the BLC in step 9 above.

After all the above processes, the five final word lists contained a total number of 17805 words of which 11838 were unique words i.e. 5967 words are duplicates among the different lists. The result of step 2 is the creation of 5 ULs – lists of single words derived from the respective dictionary entries, arranged in alphabetical order for processing purposes – one UL for each dictionary i.e. the set

$$\{UL_i \mid i \in \{BL; MW; OB; SB; WP\}\}$$

using the two-letter dictionary acronyms below e.g. ULBL denoting the word list for the BLC (Business Language Corpus). The following dictionary abbreviations are used for the remainder of the tables: MW = MoneyWords; OB = Ottawa Journal Business Dictionary; SB = Dictionary of Small Business; WP = Washington Post Business Glossary.

8.5.4 Step 3: Mapping ML_i onto UL_j

A simple matching algorithm maps each ML_i onto each UL_j. The following is the pseudo-code. Obviously the real algorithm can be made much more efficient by first sorting the word lists in alphabetical order, in which case a binary search instead of a nested loop can be performed.

```
PROCEDURE "Map ML_i onto UL_j"
FOR i  RANGING  OVER {AI; AK; AR;  ... SR ; TO; US}  /* For each model
       FOR j  RANGING  OVER {BL; MW; OB; SB; WP}/* Each dictionary
              nrmodelwords = SIZE (ML_i )              /* SIZE(X) = nr of elements in X
              nrdictwords = SIZE (UL_j )
              DIM  MAP = nrmodelwords                  /* vector same size as model list
              FOR k  =  1  TO  nrmodelwords
                     MAP_i,j(k) = 0                    /* default = no match found
                     FOR n  = 1 TO nrdictwords         /* check against each dic word
                     IF  (ML_i(k) EQUALS UL_j(n)) THEN        /* check if equal
                            MAP_i,j(k) =  1            /* found a match;
                            END IF                     /* should exit k-loop
NEXT k, n, j, i                                        /* repeat the loop
```

In the above, ML_i and UL_j are vectors of variable size containing word strings as elements. The 115 (23 x 5) vectors $MAP_{i,j}$ register the matches and contain the element values "1" or "0" depending on whether or not a match was found. (Using the values 1 and 0 is arbitrary, the strings "Found" and "Not Found" or any other Boolean values could be used as well, but using 1 and 0 facilitates calculations.) The number of elements in $MAP_{i,j}$ is the same as the number of elements of the corresponding size of ML_i. The n-th vector element in $MAP_{i,j}$ is denoted as $MAP_{i,j}(n)$.

Associated with the BLC, there are two additional vectors: BLRanking and BLWordListScore. BLRanking contains a rank value for each BLC word reflecting the frequency with which the word (or any of its word forms within the same lemma group) was found within the original million-word Business Language Corpus. This rank value ranges from "1" (for the word "the" which was found 44937 times in the corpus), "2" (for the word "to"), up to a value of "4414" for the 535 words with frequency 6 and the maximum value of "4808" for the 577 words with the minimum cut-off frequency of 5. BLWordListScore contains a "WordListScore" reflecting "the difficulty level of each word in a running text—not necessarily for the native speakers of English, but for the average adult

EFL (English as a Foreign Language) learners. All the lexical items are grouped into ten different levels of difficulty namely 01, 02, 03, 04, 05, 06, 11, 17, 21, and 30." [SOME99, p.57] A value of "01" indicates that the word is (roughly) amongst the 1000 easiest words in the English language, a value of "2" for words between 1000 and 2000 most used etc. up to a maximum value of "30" for the most difficult words. These values originate from the various source corpora and dictionaries used by a consortium of EFL researchers [SOME99].

Associated with each ML is an MLWordWeighting vector whose elements contain a weighting for each corresponding ML word entry. These weights were derived from the number of words that originally made up the entity name e.g. if the original entity name was "Gross_Operating_Income" each of the parsed words "Gross", "Operating" and "Income" would have a weighting of 1/3. For an entity name of "NetProfit" the words "Net" and "Profit" would receive a weighting of ½. The element values in the MLWordWeighting vector range from 1 to 0.125 (=1/8), since duplicate words resulting from different entity names are retained separately (in order to retain referential integrity to the original model entities).

8.5.5 Step 4: Calculation of perspicuity measures

The following perspicuity measures can be calculated.

Gross Perspicuity Count (*GPC*) for model *i* against dictionary *j* :

$$GPC_{i,j} = \frac{\sum_{k=1}^{size(ML_i)} MAP_{i,j}(k)}{size(ML_i)}$$

Note that $size(ML_i) = size(MAP_{i,j})$ = the number of original model elements (before parsing). The GPC is easiest to calculate. Its value ranges from 1 (maximum perspicuity) down to 0 for minimal perspicuity i.e. minimum perspicuity means that none of the model words are found in the dictionary j which reflects the user vocabulary. The GPC (as all of the other perspicuity measures proposed below) are normalized and can thus be expressed as a percentage. The GPC assigns an equal weight to each word in the ML. Thus, each of (matches for) the words "Net Profit Margin" of a single entity label will receive the same weighting as a single-word entity name such as "Employee".

A refinement of the GPC is the Weighted Perspicuity Count (WPC) which ensures that multi-word entity names receive the same weighting as single-word entity names:

$$WPC_{i,j} = \frac{\sum_{k=1}^{size(ML_i)} MAP_{i,j}(k) \times WW_i(k)}{\sum_{k=1}^{size(ML_i)} WW_i(k)}$$

Where WW_i represents the MLWordWeighting vector as described above. Again, the WPC ranges from 0 to 1 with a value of 1 (or 100%) representing perfect perspicuity.

Where relative word frequency is available in the dictionary, an even more sophisticated measure can be calculated by taking into account these frequencies. There are many possible ways of representing the relative word frequencies, but the most meaningful way is by means of a rank assignment within the UL. The BLRanking vector is a good example of this. The idea which underlies the formula is that each match should be weighted with the relative ranking reflecting the frequency of word use by the user. Thus a model using the more frequently used terms is more perspicuous than another model whose terminology is also found in the dictionary but uses less frequently used words. This results in a Rank-Adjusted Perspicuity Count which can be calculated both in its Gross or Weighted versions.

The following is the proposed formula for the Rank-Adjusted Weighted Perspicuity Count (for model *i* against dictionary *j*):

$$RAWPC_{i,j} = 1 - \left[\frac{\sum_{k=1}^{size(ML_i)} WW_i(k) \times \left[\{MAP_{i,j}(k) \times lookup(ML_i(k), ULRj)\} + \{[1 - MAP_{i,j}(k)] \times high_j\} \right]}{high_j \times \sum_{k=1}^{size(ML_i)} WW_i(k)} \right]$$

With:

- ULR_j the vector containing the ranking values for UL_j (with the elements ordered in the same way as UL_j). For example, ULR_{BL} = BLRanking
- lookup($ML_i(k)$, ULRj) is a function which looks up the ranking value (in ULR_j) of the word for the corresponding word $ML_i(k)$ in the ML_j. For instance, vector ML_{US} contains the word "distribution" whose rank value in ULR_{BL} is "810" (meaning it is the 810th most used word according to the BLC) so lookup("distribution", ULR_{BL}) returns the value 810. Where no match is found for the ML word, lookup can return any (finite) value, since it will be multiplied by 0 anyway.
- $high_j$ is a very high ranking to be assigned to those ML words that are not found in the UL. Possible candidate values for $high_j$ are:
    - The highest rank value within ULR_j i.e. the maximum value in ULR_j. For example, for ULR_{BL} it is 4808 i.e. the ranking for all words with frequency 5 in the BLC.
    - The next rank value i.e. the rank which would have been allocated to the next word (with the next lower frequency) in ULRj For instance, For ULR_{BL} it would be the ranking which would have been assigned to all words with frequency 4 in the BLC, namely 5385 (there are 577 words with rank 4808 or frequency 5). This choice is more defendable than the previous one, involving only a minimal extra calculation. Note that the value of $high_j$ is not equal to size(ULR_{BL})+1 since ULR_{BL} typically contains a large number of word alternative forms
    - Another similarly large number reflecting the researcher's estimate of the average ranking of the unmatched words in ML_j. An algorithm based on fitting a suitable probability distribution with the same central statistics of ULR_j can be used to estimate this, but it is unlikely that this further sophistication would increase the validity of the calculations.

When comparing a number of models, the exact choice of the $high_i$ value appears not too critical, since models that are close together will tend to have a similar number of matches (overall match count differing between 1 and 3%) and the difference between the options (especially the first two) is likely to be of the order of 10%, so the overall difference is of the order of 0.1-0.3%.

Two final notes concern the ranking algorithm. The easiest way to assign a rank is to sort words in a decreasing sequence based on their actual frequency and assign each word its position in the sorted list: the first word is the word with the highest frequency (in the corpus) and gets rank "1", etc. A problem arises for words that share the same frequency. The BLC approach was to sub-sort words within the same frequency group in alphabetical order and assign them sequential ranks. This is computationally the easiest, but methodologically not very defensible: there is no reason why "absentee" was ranked 5816th and "yield" 6407th when they both occurred 5 times in the original corpus (these are the original BLC rankings before removal of proper names etc.). The more common and intuitively most appealing approach in linguistic analysis (as in the field of competitive sports) is to assign the first rank position to all words in the same frequency group i.e. "absentee" and "yield"

(and all other words with a frequency of 5) are ranked joint 5816th; the analogy in the field of competitive sports is where, for example, three (or more) athletes cross the finish line simultaneously, they all share the first (or whatever) position. Statisticians would tend to assign the sequential position of the median word in the word frequency group to all the words in the group. For the above example, this would mean that all 591 words in the original BLC group of words with frequency 5 would share the rank of (5816+6407) / 2 = 6111½. Although the latter approach is conceptually perhaps more correct, in this research the former approach was used since it is considered to be more intuitive as illustrated by a metaphor from athletics: if the 3 front runners cross the finish line together, they are all awarded a joint first place, not an averaged second place! Also, this procedure ensures that the rank is always an integer value and is more widely used in semantic research.

A second note concerns the possible generalization of the ranking concept. Other ranking algorithms can be used, including e.g. an approach where the ranking has a much coarser granularity. This is the case for the "BLWordListScore" mentioned above, which assigns words in groups of roughly 1000 each (for the first 6000 words) or several thousands (for the less frequently used words), as explained above. Since this represents a mere scaling issue, the above formula will still work as long as the correct value for $high_i$ is used (here e.g. 30). In fact, the same formula can even be used for non-linear scale intervals as long as the measure retains order, though the interpretation loses some validity.

Finally, it is suggested that a normalized version of the RAWPC be calculated. This adjusts the RAWPC for the model size due to the fact that a larger model will naturally tend to have a higher RAWPC score than an equally perspicuous but smaller model. The extreme case is that of a model MA containing, for argument's sake, only and nothing but the 100 most frequently used (domain) words. It will have a RAWPCMA which differs from 1; because the sum of rank values 1 through to 100 is 5050, not 0. The exact value will depend on the choice of $high_i$, which is typically in turn dependent on the size of ULRj. For this example, the theoretical perspicuity should be 1 i.e. the highest perspicuity possible for a model of size 100. Furthermore, a larger model MB containing (only) the 1000 most frequently domain words should also have a perspicuity of 1 (again: it is the most perspicuous model possible) but will have a RAWPCMB which is 10 times further removed from the unit value than the RAWPCMA.

A simple adjustment factor will "normalize" the RAWPC calculation so that its value for both MA and MB cases would equal 1. The Normalized Rank-Adjusted Weighted Perspicuity Count (for model *i* against dictionary *j*) can be calculated as follows:

$$NRAWPC_{i,j} = 1 - \frac{\left[\frac{\left(\sum_{k=1}^{size(ML_i)} WW_i(k) \times \left[\{MAP_{i,j}(k) \times lookup(ML(k), ULRj)\} + \{[1 - MAP_{i,j}(k)] \times high_j\}\right]\right)}{\sum_{k=1}^{size(ML_i)} WW_i(k)} - \frac{1 + \sum_{k=1}^{size(ML_i)} WW_i(k)}{2}\right]}{\left[high_j - \frac{1 + \sum_{k=1}^{size(ML_i)} WW_i(k)}{2}\right]}$$

The interpretation of the formula is as follows: the first big fraction calculates the overall average ranking per ML concept. Reduce the average ranking by a size correction factor depending on the number of model words (i.e. the minimum possible average rank value). Divide this by the maximum possible average rank to obtain a perspicuity value in the range [0,1]. This has the overall effect of increasing the RAWPC for larger models and is important to take into account when comparing models of unequal size.

8.5.6 Step 5: Perspicuity interpretation

Table 8-6 summarizes the results of the four possible perspicuity formulae for ULR_{BL}, which is the only one which has frequency information.

The following observations can be made with respect to **validity** of the different measures:

- The increasing sophistication in formula **does significantly affect** the result: the PC values for the different models drop, on average, more than 15 percentage points from when the Gross PC is calculated to the more sophisticated NRAWPC.
- However, it is doubtful whether the additional calculations (and required lexicon information) are worth the additional effort, since **only marginal changes in *relative* positions** occur as evidenced by the relative small changes in ranking. For example, although the SAP and Silverston models swap relative positions (from 1st to 3rd), it should be realized that this represents only a very marginal difference, since they have a difference in PC of only about 0.5%.
  This is supported statistically by the fact that the model rankings based on the PCs as calculated by the different formulae, are statistically highly significantly correlated: the rank-correlation coefficients are all between 0.91 and 0.99 (with 1.00 representing perfect correlation). The corresponding z-scores are all above 4! (The detailed calculations are included in Appendix E.)

Table 8-6: Four Perspicuity Measures Using the Business Letter Corpus.

Model	GPC	WPC	RAWPC	NRAWPC	GPC rank	WPC rank	RAWPC rank	NRAWPC rank	RAWPC (WL-based)	RAWPC rank (WL-based)
AI	85.2%	86.6%	66.5%	68.2%	16	14	15	15	73.9%	16
AK	93.8%	93.3%	73.7%	75.2%	5	6	10	12	77.6%	12
AR	85.8%	86.0%	74.8%	76.5%	15	16	8	8	74.7%	15
BA	94.5%	95.1%	78.2%	81.4%	2	1	2	2	79.6%	4
BE	7.4%	5.9%	0.6%	4.2%	23	23	23	23	5.2%	23
BO	90.8%	91.8%	75.0%	77.1%	8	8	7	7	79.3%	5
CY	88.8%	88.6%	67.7%	74.1%	12	13	13	13	78.6%	8
FO	88.3%	89.1%	65.9%	67.8%	13	12	16	16	78.2%	10
HA	92.9%	92.4%	73.5%	76.4%	7	7	11	9	78.5%	9
IN	90.5%	91.5%	72.4%	76.3%	9	9	12	10	76.8%	13
MI	68.4%	68.4%	44.6%	46.7%	21	21	21	21	52.6%	21
NH	86.0%	86.6%	67.3%	70.1%	14	15	14	14	76.5%	14
NI	90.0%	91.3%	76.1%	77.9%	10	10	5	5	77.6%	11
OB	84.8%	82.3%	59.6%	63.7%	19	19	19	17	67.5%	19
OD	84.8%	82.8%	60.0%	62.8%	17	17	17	18	69.1%	17
PU	93.0%	94.5%	77.8%	79.3%	6	4	4	4	79.0%	6
RA	33.7%	35.1%	18.5%	21.8%	22	22	22	22	29.2%	22
SA	94.0%	95.0%	78.1%	81.8%	3	2	3	1	81.5%	2
SF	89.9%	90.8%	74.6%	76.3%	11	11	9	11	78.7%	7
SI	95.0%	93.3%	78.5%	81.3%	1	5	1	3	79.7%	3
SR	84.8%	82.8%	60.0%	62.8%	17	17	17	18	69.1%	17
TO	76.8%	75.7%	55.1%	59.8%	20	20	20	20	65.6%	20
US	94.0%	94.9%	75.8%	77.7%	4	3	6	6	82.1%	1
Avg	**82.3%**	**82.3%**	**64.1%**	**66.9%**					**70.0%**	
StDev	**20.8%**	**21.0%**	**19.4%**	**19.2%**					**18.3%**	

- Additionally, the overall discriminatory power of the metric does hardly change, whether a straightforward GPC or a complicated NRAWPC is calculated, since the standard deviation between the PC scores remains a relatively constant 20%.
- However, where possible – especially where the calculations can be automated – the more sophisticated formulae remain the preferred ones from a scientific and conceptual point of view. This is confirmed by the fact that the larger models indeed benefit from the additional sophistication: the relatively larger models of Hay, Inmon, Ottawa-Big and SAP all improve their relative standing from RAWPC to NRAWPC. Interestingly, most of the changes in relative position from RAWPC to NRAWPC tend to "undo" some of the (often relatively big) changes introduced by the change from WPC to RAWPC: e.g. Hay "lost" 4 positions (from WPC to RAWPC) but gains back 2 (from RAWPC to NRAWPC), Inmon lost 3 and then gains back 2; Silverston gained 5 but then loses 2.
- The change from an unweighted (GPC) to a weighted measure (WPC) appears to have remarkably little influence overall or for individual models.
- The biggest change in ranking and values occurs when moving from WPC to RAWPC. It is unclear whether the move to take into account relative word frequencies is validated on face value by looking at the results. Interestingly, there are huge effects for some models. For example, the ARRI and Nippon models move up dramatically. Is this because they are models drawn up by English-as-a-foreign-language speakers and hence use simpler, more frequently used, words? Could it be argued that the AKMA, Fowler, Hay and Inmon have been drawn up by consultants, who use a more sophisticated vocabulary?
- An important validation issue is that the BelgAcc, Random and Miller models all obtain very low scores. Indeed, the BelgAcc model should score very close to 0 (as it does). The Random model with its "random English words" scores equally low, and so it should, since it does map onto the business language. However, it is probably more perspicuous than the PC suggests, since users of business language are apt to be equally skilled in common English. Finally, the low score of the Miller model is also vindicated by its use of systems theory or even made-up words. At the other extreme, it is good to note that most of the commonly accepted and used models do indeed use good English business terminology, as reflected in the relatively high overall scores.
- Finally, the ranking can also be calculated using WL ratings instead of more detailed word frequencies. As could be expected, the WL-based rankings are slightly different and in fact move closer to the less valid lexicon ratings below. It appears certainly advantageous to use the additional resolution or precision of absolute word frequencies, although WL ratings are better than no frequency information at all (as shown below).

Having validated the formulae, it is time to analyse the specific models:

- At the top end, there is a tight, high-scoring cluster of models consisting of SAP, Baan and Silverston, all scoring between 81.8% and 81.3%. Is it coincidence that two of these are leading ERP systems used in businesses all over the world? Can it be argued that an 80% implies outstanding perspicuity?
- Following very closely behind, but with a distinct gap, is a group of second-tier models in the narrow band between 79.3% and 74.1% consisting of Purdue, Nippon, USB, BOMA, ARRI, Hay, Inmon, SanFran, AKMA and CYC, who can all be said to have excellent perspicuity with a suggested cut-off of roughly 75%.

- Fairly close, but with a distinct gap, are NHS with 70%, which uses some non-business terminology due to its medical bias, and both AIAI and Fowler who, at 68%, both use some non-regular business terminology. The 65% to 75% range can therefore be said to be indicative of possible problems.
- Low scorers are the Ottawa and TOVE models, which both use too much non-regular language e.g. the Ottawa-derived models (including Semi-Random) have a large number of specialized accounting and financial terms.
- As mentioned above, a number of models have severe perspicuity problems – as was to be expected.
    - BelgAcc is developed in another language and should therefore be tested against a different (Dutch) word list.
    - Random was specifically designed to have a low semantic meaning.
    - Miller is recognized as using non-standard terminology and would be a hard sell to the business community.

The above analysis corresponds with the intuitive analysis of the model entity names, in turn providing additional validation to the measure. Table 8-7 lists the WPC scores for the other user lexicons.

Table 8-7: Weighted Perspicuity Counts and Rankings For Other Lexicons.

Dict (UL) / Model (ML)	Weighted Perspicuity Count (WPC)						WPC Ranking					
	BL	MW	OB	SB	WP	All	BL	MW	OB	SB	WP	All
AI	86.6%	82.0%	47.3%	60.5%	45.2%	91.1%	14	11	13	14	13	16
AK	93.3%	81.1%	42.7%	53.8%	40.9%	95.9%	6	14	15	16	15	7
AR	86.0%	74.7%	30.3%	60.3%	32.2%	87.4%	16	17	19	15	18	19
BA	95.1%	82.9%	59.8%	72.1%	55.9%	96.6%	1	9	5	8	5	5
BE	5.9%	3.5%	1.6%	2.0%	1.6%	7.3%	23	23	23	23	23	23
BO	91.8%	81.4%	47.0%	65.0%	44.0%	93.4%	8	13	14	11	14	15
CY	88.6%	63.6%	32.5%	49.1%	32.1%	90.9%	13	18	18	18	19	17
FO	89.1%	79.5%	42.0%	52.6%	50.4%	95.3%	12	16	16	17	10	10
HA	92.4%	83.0%	49.8%	60.5%	52.5%	94.8%	7	8	11	13	8	12
IN	91.5%	82.2%	59.4%	75.2%	53.5%	94.9%	9	10	6	5	7	11
MI	68.4%	41.5%	22.2%	31.2%	13.9%	71.2%	21	21	21	21	21	21
NH	86.6%	59.5%	34.1%	34.6%	37.8%	88.4%	15	19	17	20	17	18
NI	91.3%	80.4%	49.8%	73.2%	39.4%	94.6%	10	15	12	7	16	13
OB	82.3%	93.0%	100.0%	87.7%	68.1%	100.0%	19	4	1	4	4	1
OD	82.8%	95.2%	100.0%	92.0%	73.7%	100.0%	17	1	2	1	2	1
PU	94.5%	84.6%	50.8%	73.6%	49.4%	95.8%	4	7	10	6	11	8
RA	35.1%	26.0%	7.7%	11.9%	7.1%	40.7%	22	22	22	22	22	22
SA	95.0%	81.5%	51.1%	67.0%	46.7%	95.7%	2	12	9	9	12	9
SF	90.8%	85.1%	58.1%	66.6%	53.9%	93.9%	11	6	7	10	6	14
SI	93.3%	85.7%	54.3%	61.2%	51.6%	96.4%	5	5	8	12	9	6
SR	82.8%	95.2%	100.0%	92.0%	73.7%	100.0%	18	1	2	2	2	1
TO	75.7%	51.4%	24.9%	37.8%	30.3%	79.6%	20	20	20	19	20	20
US	94.9%	93.3%	85.2%	90.0%	81.3%	97.9%	3	3	4	3	1	4
Avge	82.3%	73.3%	50.0%	59.6%	45.0%	87.0%						
StDv	21.0%	23.0%	26.7%	23.8%	20.1%	21.5%						

Note that MW, OB, SB and WP are less validated lexicons, i.e. not academically researched and derived from a large language corpus. For comparative reasons, the WPC for the BL has also been included.

Firstly, it is important to repeat that there is a significant overlap or correspondence between the last 4 ULs (as shown in Appendix E). Also, one should realize that these ULs tend to have a more financial and accounting bias.

- The "severely problematic" models, remain the same: BelgAcc, Random and Miller score unacceptably low, in fact even more pronounced than for the earlier measure.
- The low-scorers remain NHS, AIAI, TOVE and Fowler but they are joined by the previous high-flyer ARRI and marginal CYC and AKMA, whilst the Ottawa-derived models (OB, OD and SR) all move up to the highest position. This is due to the model selection/design (business lexicon derived) and must therefore be ignored for this analysis (although it validates the measures).
- When the Ottawa models are excluded for methodological reasons, the previous top-scorers appear to merge with the second-tier models forming a fairly close clump consisting of Baan, Silverston, SAP, Inmon, Purdue, SanFran with BOMA relatively marginally at the bottom.
- The only distinct move is from the USB model which has very high scores. This can be attributed to the fact that it is a financial model and indeed uses terminology reflected in the ULs.

From this analysis, it appears that using the remaining ULs as proxies to the BL is subject to the following validity concerns:

- As lexicographers have known, there is a distinct bias in dictionaries and they do not necessarily reflect spoken language. In particular, the 4 lexicons used here have a strong financial/accounting bias.
- The PCs seem to lose a fair bit of discrimination power when less validated lexicons are used. There is no distinction between the "excellent" and "good" models.
- Of the lexicons used, only the MoneyWords UL seems to provide reasonable scores. The remaining ones average quite low perspicuity scores across most models and also show markedly increased standard deviations. Increased variability is often indicative of problematic validity.

From this, it can be concluded that the use of a well-validated lexicon derived from a large corpus and including word frequency statistics, combined with the hopefully automated calculation of the more sophisticated PC, is highly recommended and will yield valid results. Using a dictionary-based word-list instead is likely to result in very non-discriminatory or variable scores, and will highlight only patently problematic perspicuity (in which case there is no need to go to all the effort of calculating the PC) or the PC may be influenced by biases in the model selection.

Note that, in the absence of a well-validated lexicon, it *is still possible* to use other lexicons. Table 8-8 provides statistical support for the fact that the rankings produced by the different lexicons correlate in fact fairly well: all the z-scores are statistically highly significant.

Table 8-8 gives the the rank correlation coefficient and the z-score (significance) for the different possible pairs of lexicons for which the relative model rankings are compared. In all cases, the Z-score is above the 1.96 required for 5% significance, although only very (marginally) so when one swaps the BL for one of the other lexicons. Note that, for the correlations involving the BL lexicon, the Ottawa-derived models were included from the calculations. If they are excluded, in the light of the methodological problem explained above, the Z-scores increase substantially. The full table with

the relative contributions of each model to the overall rank-correlation coefficient is included in Appendix E and is useful to see which models have changed their relative position most when changing respective UL.

Table 8-8: Correlation Between Perspicuity Rankings Using Different Lexicons.

	BL -> MW	BL -> OB	BL -> SB	BL -> WP	MW -> OB	MW -> SB	MW -> WP	OB -> SB	OB -> WP	SB -> WP
Correlation coeff. r' =	0.43	0.44	0.44	0.45	0.95	0.89	0.94	0.94	0.95	0.87
Significance Z-score z =	2.02	2.06	2.06	2.09	4.45	4.19	4.43	4.39	4.47	4.08

8.6 Quality of Documentation

Although the quality of the documentation can be seen as directly related to the overall perspicuity of the model, it is normally considered under a separate heading in most evaluative frameworks as discussed in Chapter 5.

8.6.1 What is documentation?

A wide interpretation of documentation includes any information that assists with the interpretation of the model can be seen as model documentation. This includes notes on the context, interpretation, implementation and modification of the model. A narrower interpretation of documentation looks at how well each of the individual model elements is defined *within* the model, using human-readable language i.e. does the model provide definitions or descriptions for each element which has been modelled?

When dealing with a small number of models, it will be preferable to compare or evaluate all model documentation i.e. in its "widest" interpretation. This will include a wide variety of sources including tool-based documentation such as online help, web-based documentation, various manuals, industry reviews, and academic publications. For example, the "documentation" of the SAP R/3 reference model may arguably include any of the reference books on SAP, but does it also include such remarks as the most suitable type of corporate model around which current ERP systems have been built [e.g. TODD98]? In a way, a lot of this documentation can be seen as meta-model information.

The narrow definition is more amenable to consistent interpretation and delineation across different disciplines, and will therefore be favoured in this research. It can also be described as internal documentation since it is directly included in our meta-model and forms an integral part of the model database.

8.6.2 Attributes of documentation quality

The following are some suggested attributes of documentation quality which can be measured fairly easily:

- ***Completeness***: the proportion of model elements defined. Ideally all model element instances should be defined. In practice, most static models will provide definitions for the model entities but not necessarily for their relationships.
- **Inter-linkedness**: the degree to which definitions cross-referenced (or hyper-linked) i.e. the inter-linkedness between model elements. The use of standardized terminology provides cognitive

coherence, and thus it can be argued that the cross-reference of concepts facilitates model comprehension. A formal way of cross-referencing also enforces consistent use of terminology.

- **Extensiveness or depth**: the amount of detail or description provided for each element. The more explicit or comprehensive the definitions, the better.
- **Examples**: whether real world examples or sample instances are provided or not.
- **Readability of the descriptions or definitions**: ideally, the readability should be measured against the language used by the intended audience. Hence this refers back to the perspicuity analysis (or the perspicuity analysis could be extended to include the documentation's terminology). In practice, users will often refer to the documentation where the perspicuity of the original model element name, i.e. the business or domain language, is unclear, hence a more universal or generic readability analysis should be attempted.

8.6.3 Completeness

Most models define at least some of their model element instances. Only three models do not include any documentation: the BelgAcc, Inmon and Nippon models. It must be noted that the interpretation of some of the elements in the BelgAcc is provided within the Royal Decree as well as the Notes of Interpretation that have been issued subsequently. However, these have been ignored in our analysis since there is no easy systematic way of integrating these; they are not in English and cover only a small fraction of the model anyway.

Some models, typically those not captured from an electronic source but from a book, do not provide explicit definitions as an intrinsic part of the model. However, they do provide another, usually less systematic, way of defining some or all of their model element instances, which is usually what the book is about.

- Scheer introduces virtually all of the model elements in the text and provides a subject index.
- The Baan book generally does not provide detailed descriptions on the model elements but it does have an index.
- Silverston describes most model elements in the text, has an index and a separate listing of model entities with their attributes (and attribute types).
- Fowler defines some of his model elements in the text and has an index.
- Miller gives descriptions of his major model elements in a summary table, along with typical examples.

Table 8-9 gives an overview of the degree of the completeness of the documentation for the models mentioned (in its wide sense).

The *Overall Completeness* (OCB, i.e. based on documentation in book form) is a suggested composite index calculated as follows:

$$OCB = [Index\#] + [Text\#] + [Glossary\#]$$

Where:

- [Index#] = 2 if the documentation has an index which includes the model element names; = 0 if not.

- [Text#] = 2 if the documentation explicitly defines or explains all (or the large majority of) the model elements in the text; = 1 if *some* of the elements are defined; = 0 if no or very few elements are formally defined.
- [Glossary#] = 2 if the documentation includes an alphabetical list with definitions of terms/model elements used; = 1 if a glossary with *some* model elements is included; = 0 if no glossary is included.

Table 8-9: Degree of Documentation Completeness for Book-based Models.

Model ID	Model Code	Index	Text	Glossary	OCB	Overall Completeness
Miller	MI	Yes	Yes	Some	5	Excellent
Fowler	FO	Yes	Yes	No	4	Very good
Hay	HA	Yes	Yes	No	4	Very good
SAP	SA	Yes	Yes	No	4	Very good
Silverston	SI	Yes	Yes	No	4	Very good
Baan	BA	Yes	No	No	2	Low
BelgAcc	BE	No	Some	No	1	Very low
USB	US	No	Some	No	1	Very low
Inmon	IN	No	No	No	0	Nil
Nippon	NI	No	No	No	0	Nil

To the right is the subjective interpretation of the index with the suggested value mapping between the OCB index and the interpretation of 6 = Perfect; 5 = Excellent; 4 = Very good; 3 = Good; 2 = Low; 1 = Very Low and 0 = Nil.

A completeness index for models that include model element definitions within the model itself (the narrow interpretation of documentation) can be calculated as follows:

OCM = Number of Model Elements Defined / Total Number of Model Elements

i.e. the relative proportion of model elements that are defined, best expressed as a percentage. The only model that gives explicit definitions for its relationships is the AIAI model. None of the models defined their groupers, although it can be argued that a grouper is sufficiently defined by its constituent elements. Computing the OCM for models that do not define the relationships (or groupers) gives a very distorted view of the documentation completeness due to the high variability in the relative proportion of relationships (and groupers) in the various models. Hence a modified version of the above formula is suggested where only the entities in each model have been considered:

OCE = Number of Model Entities Defined / Total Number of Model Entities

Table 8-10 lists the values for each model.

Table 8-10: Completeness Ratings for Models That Include Entity Definitions.

Model ID	Model Code	OCE	Completeness
Random	RA	100%	Perfect
AIAI	AI	100%	Perfect
CYC	CY	98%	Excellent
ARRI	AR	98%	Excellent
Ottawa-Big	OB	98%	Excellent
AKMA	AK	98%	Excellent
Semi-Random	SR	97%	Excellent
Ottawa-Dense	OD	97%	Excellent
NHS	NH	80%	Good
BOMA	BO	74%	Good
Purdue	PU	71%	Good
SanFran	SF	50%	Fair
TOVE	TO	20%	Low

Only the Random model and the AIAI model provide explicit (English) definitions for all of their entities. Note that the entities in the Random model are drawn from the Oxford Paperback Dictionary so its inclusion in this analysis is not really applicable. Where the OCE is nearly 100%, the difference must often be explained with technical reasons e.g. abstract super-types.

The only really perfectly complete model is AIAI since it also includes definitions for its relationships. Most of the other models have excellent definition coverage, with the SanFran and TOVE models scoring lowest and a middle group consisting of NHS, BOMA and Purdue.

8.6.4 Inter-linkedness

The easiest way of formally ensuring lexical consistency is by using explicit hyperlinks in the definitions. Formal models such as CYC, AIAI and TOVE make good use of these, as do on-line-lexicons (i.e. the Ottawa models, which also form the basis for the Semi-Random model). The metric calculated in Table 8-11 is the (mean) average number of explicit hyperlinks in the definitions. Each model uses a different method to indicate hyperlinks:

- The Ottawa model uses CAPITALIZATION. Note that the Ottawa model also includes 416 cross-references to concepts that are not (yet?) defined in the lexicon. (Semi-Random uses the same lexicon and definitions as Ottawa-Dense.)
- The CYC model uses the #$-prefix for its entity names. Note that a number of hyperlinks point to entities outside the "generic enterprise" subset selected for this research. If links to non-enterprise related entities are included, the average number of hyperlinks per definition jumps to 5.3, by far the highest value of all models.
- TOVE mixes common English words as well as specially defined entity words in its definitions. The only unambiguous hyperlink in the textual definitions is the use of variables, which are formatted with the question mark symbol "?" as a prefix.
- AIAI uses a clear and consistent naming of entities in its definitions in the form of "initial capitalized words", concatenated with hyphens e.g. "Execution-Of-Activity-Spec".
- NHS uses double quotes for references to other entity names within definitions.

Because of the nature of the above models, the average number of hyperlinks is closely related to the syntactic complexity, and this metric therefore does not appear to add much value in analysing the quality of the documentation.

Table 8-11: Hyperlinks in and Extensiveness of Entity Definitions.

Model ID	Model Code	Average nr of hyperlinks /definition	Extensive-ness (Average)	Extensive-ness (Median)	Examples
Ottawa-Dense	OD	1.8	511	440	No
Semi-Random	SR	1.8	511	440	No
CYC	CY	1.8	443	364	No
Ottawa-Big	OB	1.4	423	348	No
SanFran	SF		357	289	Most
AKMA	AK		328	225	Most
Purdue	PU		187	181	No
ARRI	AR		303	173	No
BOMA	BO		237	171	Few
TOVE	TO	2.3	160	153	No
NHS	NH	2.2	165	108	No
AIAI	AI	2.8	116	81	No
Random	RA		111	71	Some

8.6.5 Extensiveness

Relatively easy to formalize for *all* models is the extensiveness of the documentation. Although one might enumerate a model's total size (in bytes, words or pages), obviously the model size or number of elements influences the absolute size of the documentation. Table 8-11 lists the average size of each entity definition in characters. Since definition lengths exhibit rather skewed distributions, it might be suggested that the median should be used rather than the mean average. However, the table (sorted in descending order of median definition length) shows that the ranking is fairly insensitive to the choice of average. Only the Purdue model (which has by far the most skewed distribution of definition lengths), and to a smaller extent the NHS model, change their relative position.

A difference of 50 characters or so should not materially affect the quality of the documentation. Although no absolute standard can be set, the following rough guidelines are suggested:

- Sparse documentation: average definition length less than 140 characters (1 or 2 lines of text): the Random, AIAI and NHS model.
- Extensive documentation: average definition length exceeds 350 characters (more than 5 lines): Ottawa and CYC models.
- Adequate documentation: the remaining models.

8.6.6 Use of examples

The only models that make extensive use of examples as part of the entity definitions to illustrate the model entities are the SanFran and AKMA models, though BOMA includes occasional examples. Especially in the context of *generic* enterprise models, the use of examples can be considered an important aspect of documentation. It is therefore most unfortunate that most of the models fall short in this respect.

8.6.7 Readability index

The readability of the documentation is very important, especially when considering the fact that the documentation is often consulted when the original model element (entity) name is unclear or imprecise. Although the relationship between model perspicuity and readability of documentation was pointed out earlier, it is here considered that the readability of the documentation must not necessarily be measured against a domain-specific vocabulary.

There are many ways to measure readability of text, though the favoured approach is by means of readability formulae. Their popularity can be explained because of the objectivity and ease of computation. [FLES79]. There exists a large number of these readability formulae, at least 200 by a popular count, though only a few are commonly computed e.g. Gunning's FOG test, Fry's readability graph, Flesch-Kincaid's formulae, Powers-Sumner-Kearl's formula, McLaughlin's SMOG formula and the US army's FORCAST formula [JOHN87].

A number of stand-alone text analysis computer programs exist to calculate various readability indices, though their use is somewhat obviated by modern word processing software which includes tools to generate readability (and other text statistics) automatically.

Microsoft's Word 2002 program was used to generate the following two readability indices (refer to MS-Word online help file and [FLES79]).

8.6.7.1 The Flesch Reading Ease score

The FRE rates text on a 100-point scale; the higher the score, the easier it is to understand the document. For most standard documents, it is suggested that one should aim for a score of approximately 60 to 70 *in general writing*. The formula for the Flesch Reading Ease score is:

$$FRE = 206.835 - (1.015 \times ASL) - (84.6 \times ASW)$$

where:

- ASL = average sentence length (the number of words divided by the number of sentences)
- ASW = average number of syllables per word (the number of syllables divided by the number of words)
- The highest allowable value of FRE is 100 and the lowest value 0.

8.6.7.2 Flesch-Kincaid Grade Level score

The FKGL rates text on a U.S. grade-school level. For example, a score of 7.0 means that an seventh grader can understand the document. For most standard documents, aim for a score of approximately 7.0 to 8.0. The formula for the Flesch-Kincaid Grade Level score is:

$$FKGL = (.39 \times ASL) + (11.8 \times ASW) - 15.59$$

With ASL and ASW as defined above but with MS-Word allowing a maximum value of 12 and a minimum value of 0. Figure 8-2 gives a screenshot of a typical MS-Word readability statistics report

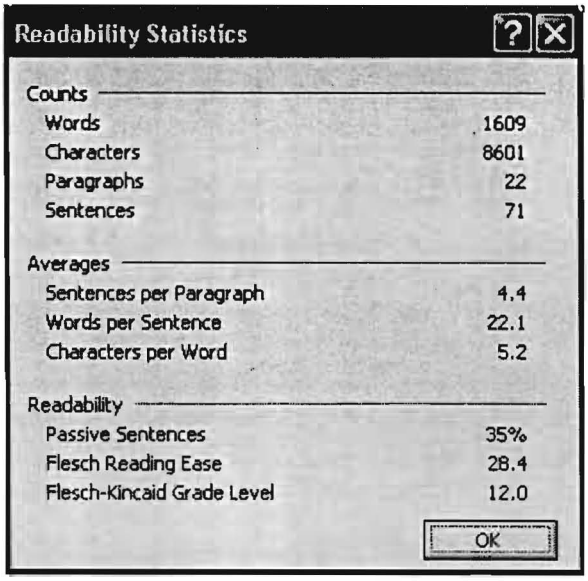


Figure 8-2: Sample MS-Word Readability Statistics Report (BOMA model).

The readability analysis was applied to the entity definitions where available. For the other models, sample pages of continuous text from the book describing the model were scanned and analysed. To get a fair representation of each book, two consecutive full pages of text nearest to each quartile division in each book were scanned. For example, for a book of 398 pages of text (excluding index and appendix), the candidate sample pages would be pages 99 & 100; 199 & 200; 299 & 300. Often, any of these (pairs of) pages contained some large diagram or table in which case two consecutive full-text pages as near as possible to the candidate pages would be chosen instead. From the scanned text, any partial paragraphs were deleted since this might have affected some of the text statistics.

Table 8-12 lists the readability scores. The full set of readability statistics can be found in Appendix I. The following interpretations can be given in comparing the different models:

- Most documentation – especially that consisting of model definitions – has a very low readability as evidenced by the maximal FKGL values (grade 12 level) and low FRE scores.
- The Purdue model has the lowest readability score. This is partly due to the lack of punctuation (full stops) in its descriptions. Consider the following typical example: "Cost Balancing And Budget: Establishment of Criteria and Tests to Assure That Operational Budget is Being Followed, Collection of Raw Material, Labor, Energy and Other Costs for Transmission to Accounting". As a sensitivity check, it was found that separating the definitions in several sentences did not necessarily increase the Purdue scores significantly. The first page of (14) definitions was re-written by cutting the definitions into short sentences. This caused only a minute increase in FRE from 1.3 to 1.5; and the FKGL remained at grade level 12. The low readability score is confirmed by the fact that the Purdue model has the highest average word-length, namely 6.2 characters per word.

Table 8-12: Model Readability Scores.

Model ID	Model Code	Source	FRE	FKGL
Hay	HA	Book	48.0	11.1
USB	US	Book	45.4	10.6
Silverston	SI	Book	45.0	11.8
Fowler	FO	Book	41.2	11.4
Baan	BA	Book	34.9	11.8
Miller	MI	Book	23.7	12.0
SAP	SA	Book	16.9	12.0
Random	RA	Definitions	68.1	5.4
TOVE	TO	Definitions	53.1	11.9
BOMA	BO	Definitions	42.3	11.7
SanFran	SF	Definitions	37.7	12.0
AKMA	AK	Definitions	34.6	12.0
Ottawa-Big	OB	Definitions	34.4	12.0
Ottawa-Dense	OD	Definitions	33.6	12.0
Semi-Random (=OD)	SR	Definitions	33.6	12.0
NHS	NH	Definitions	33.5	12.0
CYC	CY	Definitions	31.8	12.0
AIAI	AI	Definitions	23.8	12.0
ARRI	AR	Definitions	17.3	12.0
Purdue	PU	Definitions	1.3	12.0

- The next lowest two FRE scores were obtained by the SAP and ARRI models. Although this may be a coincidence, it so happens that these are both models originating from non-English speaking countries: Scheer's SAP book was translated from German, the ARRI model was developed in Italy. The low readability score is again validated by the average word length: the ARRI model has the second-highest (5.9) and the SAP model the third-highest (5.7). Notably, the Baan model ties in respect of average word length with Baan (also 5.7) but, according to MS-Word's analysis, 53% of the sentences in the SAP model are passive sentences (the highest value of all models), as against none (0%) of the sentences in the Baan documentation. This may again be seen as an additional validation of the FRE index, since the percentage of passive sentences is not included in the computation of either of the two readability scores.
- The best readability score was obtained by the Random model. This can be explained by the fact that its entities (and definitions!) were drawn from the Oxford dictionary, which is aimed at a completely different, less sophisticated market, than the other "regular" enterprise models.
- Apart from the Random model, none of the models has an FRE readability index which is even close to the "suggested value" of 60 but, as intimated above, the general enterprise models are generally aimed at a more sophisticated public with graduate qualifications.
- The model with the most readable definitions appears to be the TOVE model, with the BOMA model coming a clear second. The remaining models are all clustered rather closely within the narrow FRE score band of 31.8 to 37.7. The models whose documentation are books tend to have slightly higher FRE scores.
- It is interesting to note that the USB FRE score of 45.4 was calculated using the full user instruction manual (since it was available in electronic format). For comparative purposes, the text portion of the programmer's reference manual was analysed and found to have an almost identical FRE score of 45.2, although the FKGL increased by a half grade from 10.6 to 11.1.

How valid are the readability measures? The following observations need to be considered:

- There appears to be a fairly strong correspondence between FRE and FKGL, especially for the definitions. This confirms the validity of the two measures (although it must be recognized that they are both based on a linear combination of the same variables).
- The FKGL appears not to be a very good discriminator: too many models have equal scores i.e. the maximum score. It may therefore be advisable to use the FRE as a guide. In Table 8-12, models are sorted according FRE value.
- There are serious validity issues when comparing the readability of the definitions directly with the readability of the text samples from the books. For instance, for the BOMA model, both the readability of the book (FRE=28.4; FKGL=11.7) and the definitions (FRE=42.3; FKGL=12.0) were calculated and show a fairly big difference. If the book FRE score was used instead of the definition FRE score to rank the BOMA model against the other models in the "definitions" list, it would drop 8 positions from third-from-the-top to third-from-the-bottom.
- An apparent anomaly is the low readability score of the Baan documentation, as opposed to the high perspicuity scores. However, it must be remembered that the book was written by completely different people than the Baan model itself, so the two cannot be expected to correlate. In most other cases, there seems to be some degree of correlation between model perspicuity and the readability of the documentation, adding to the overall validity of the measure.

Overall, it appears that calculating the FRE readability score is a useful analysis, as long as it is remembered that model documentation is targeted at a relatively sophisticated public. Hence, FRE scores should not be expected to range in the sixties or above, but special note should be taken if the score falls below 30. It would be a useful validation exercise to have a panel of English communication experts assess extracts of the model documentation and see if their opinion confirms the actual readability scores.

8.7 Relative Model Correspondence – Similarity and Cluster Analysis to Measure Domain Overlap

A very interesting analysis is the problem of how closely related the various models are to each other i.e. how much of their domain overlaps. Because this entails looking at what is being modelled, i.e. model meaning, it falls under the category of semantic analysis.

In principle, the approach to measuring this model correspondence or model overlap is simple: attempt to map the various model constructs of each model to a semantically equivalent model construct of each of the other models. Wherever corresponding model constructs are found, these increase the similarity (or degree domain overlap) between the models; the more model elements that are found for which no corresponding element can be found in another model, the more this increases the difference (semantic distance) between the models concerned.

Although the correspondence analysis could – and should – in principle be applied to all model elements, including relationship and groupers, this was not done due to the fact that too many models in the database did not name or describe their relationships or grouper constructs. As before, a pragmatic methodology decision was taken to restrict the analysis to measuring the overlap between model entities only, although some model structure in terms of super/subclass was taken into account.

Another approach was used in [CHEN94 and CHEN98] where a "heuristic similarity assessment function" was used to "quantify the quality of a match" between entities in their "Generic Model

Advisor". Their function includes the matched structural relationships instead of attributes but depends on specific user interaction to identify matches. The approach suggested in this thesis, in contrast, can be fully automated (no user interaction is required) and can have a more sophisticated approach in terms of mapping of similarity using meaning rather that structural similarity.

The overall methodology can be broken down into three distinct steps, each posing unique challenges:

1. Mapping the entities from each model to the corresponding entities in other models. This involves semantic processing, and results in large tables with cross-linked concepts.
2. Based on the mappings, calculate the similarity (or distance, which is the equivalent) between the various models. This involves choice of distance measures and a large number of calculations, typically resulting in a matrix-like table with distance.
3. Analyse and interpret the similarity indices. The distance tables represent a multi-dimensional space (22 dimensions for 23 models) and are difficult to interpret at first sight. Further statistical transformations are required to visualize the findings.

Note that a somewhat similar approach was suggested in [HONK98] to investigate the similarity between documents, although a much more refined approach will be presented below, in order to deal with synonyms.

8.7.1 Mapping the Semantic Correspondence between Entities.

The first step involves analysing each model entity and checking whether it can be mapped to an equivalent entity in another model. Although domain experts are fairly good at deciding whether an entity in model A corresponds to another entity in model B, in practice this is a fairly subjective process, depending on accurate names or description of model entities as well as the skill of the domain expert. Again, it is important to reiterate the *caveat* mentioned in section 8.5.1: the use of the same word (identical linguistic tokens) in two different models does not necessarily imply that the respective modellers meant the same concept, since different persons ascribe different meanings to words and many words have multiple meanings (i.e. are "semantically overloaded") even within restricted domain contexts. Again, this methodological problem is acknowledged but pragmatic considerations force the use of word tokens rather than underlying meanings.

Much more problematic from a theoretical point of view is that the concept mapping process become computationally intractable as model size increases. Semantic entities cannot be ordered the same way that numbers or values are: alphabetical listing does not help because synonyms can be spelled very differently. Hence the computational complexity of the mapping algorithm is of the order of N^2, increasing rapidly as model size grows. Admittedly, hierarchical structuring of semantic concepts is possible, reducing the computational complexity significantly. Nevertheless, the emphasis of this research is on finding measures which can, in principle, be automated.

The first, naïve approach is to look for entities with the same name: most enterprise models use common domain names for entities such as "customer", "payment", "worker" etc. Again, instead of using the original entity names (labels), the standardized single word lists for each model which were developed for the perspicuity analysis were used.

A relatively severe methodological shortcoming here is the fact the meaning of composite multi-word entity names typically does not correspond to the meaning of the constituent single words. This was not seen as a serious obstacle in the perspicuity analysis but the argument is much more severe in the context of the similarity (as well as domain coverage) analysis. The WordNet linguistic database used below does indeed incorporate composite word entries. An attempt was made to use the *full* i.e.

composite entity names but it was found that only 227 out of 3868 unique composite entity names (i.e. less than 6%) were found to exist as separate entries in the WordNet lexicon. In fact, only 94 composite entity names appear in more than one model (excluding Semi-Random and Ottawa-Dense whose entities are a sub-set of Ottawa-Big). This forced the use of the single word entity lists since the alternative would have left too many entities out or, alternatively, it would force the use of prohibitively expensive manual procedures. Some comfort can be found in the fact that the methodological problem is not quite as bad as it seems at first sight since inspection reveals that the meaning of many composite entity names is indeed relatively close to the meaning of the constituent words e.g. "abandoned action" (Fowler) and "abandoned activity" (NHS). Nevertheless, it is hoped that future research will find a methodologically more acceptable way of dealing with the conundrum.

In order to measure model overlap, a "matching count" was done for each word in each $ML_i \mid i \in \{AI; AK; AR; BA; BE; \ldots SR ; TO; US\}$ using the two-letter model acronyms e.g. ML_{AI} denoting the word list for the AIAI model.

Because no weighting is to be applied, a more efficient matching algorithm than that for the perspicuity analysis can be used to map each ML_i onto each UL_j. The following is the pseudo-code. Obviously the implementation of the algorithm can be made much more efficient if the word list is sorted in alphabetical order, in which case a binary search instead of a nested loop can be performed.

```
PROCEDURE "Map ML_i onto ML_j"
DIM Count(23,23)                                    /*Matrix holds count of matches
FOR i RANGING OVER {AI; AK; AR; ... SR ; TO; US}    /* For each model
        NrmodelAwords = SIZE (ML_i )                /* SIZE(X) = nr of elements in X
        FOR j RANGING OVER {AI; AK;. ... ; TO; US}  /* Each model
                NrmodelBwords = SIZE (ML_j )
                Count_i,j = 0                       /* initial count = 0
                FOR k = 1 TO NrmodelAwords
                        Foundmatch = False          /* initially no match found
                        FOR n = 1 TO NrmodelBwords  /* check each model word
                        IF (ML_i(k) EQUALS ML_j(n)) THEN
                                FoundMatch = True   /* found a match;
                                END IF              /* should exit the k-loop
                        Next n                      /* check all ModelB words
                IF FoundMatch THEN Count_i,j++      /* increase count by one
                Next k                              /* check next ModelA word
        NEXT j                                      /* check next modelB
NEXT i                                              /* do all models (ModelA)
```

Table 8-13 gives the raw counts of matches for each model against each other model i.e. a table of 23 by 23. Note that this table is symmetrical along its diagonal i.e. can be transposed without changing any values.

Table 8-13: Model Overlap - Number of direct matches found between entity names of models.

Model	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US	
AI	94	8	14	24	1	20	39	15	26	19	3	12	13	24	14	10	4	19	13	18	14	19	5	94
AK	8	83	10	22	1	18	38	10	27	31	5	14	10	18	11	8	3	23	17	22	11	11	7	83
AR	14	10	147	33	1	25	35	12	25	31	9	12	45	21	13	25	3	41	13	17	13	22	6	147
BA	24	22	33	258	2	61	91	36	65	87	9	30	44	68	43	31	10	83	50	64	43	44	36	258
BE	1	1	1	2	349	1	1	1	1	2	0	0	2	3	2	1	0	1	1	1	2	2	0	349
BO	20	18	25	61	1	174	67	25	54	59	8	12	28	51	34	23	3	62	32	51	34	27	20	174
CY	39	38	35	91	1	67	697	43	66	99	12	49	43	90	59	26	15	81	55	58	59	71	40	697
FO	15	10	12	36	1	25	43	113	30	31	0	31	10	29	22	13	4	32	28	30	22	19	13	113
HA	26	27	25	65	1	54	66	30	227	72	6	33	42	54	34	37	10	68	27	56	34	31	22	227
IN	19	31	31	87	2	59	99	31	72	411	11	24	49	123	93	41	10	78	40	63	93	33	49	411
MI	3	5	9	9	0	8	12	0	6	11	60	1	10	10	7	4	1	11	5	6	7	9	4	60
NH	12	14	12	30	0	12	49	31	33	24	1	177	12	21	11	5	4	30	16	23	11	26	10	177
NI	13	10	45	44	2	28	43	10	42	49	10	12	196	39	22	45	5	55	12	20	22	28	12	196
OB	24	18	21	68	3	51	90	29	54	123	10	21	39	451	277	31	10	60	37	46	277	31	68	451
OD	14	11	13	43	2	34	59	22	34	93	7	11	22	277	277	17	7	35	24	30	277	15	57	277
PU	10	8	25	31	1	23	26	13	37	41	4	5	45	31	17	129	2	37	12	25	17	11	12	129
RA	4	3	3	10	0	3	15	4	10	10	1	4	5	10	7	2	249	6	1	4	7	4	2	249
SA	19	23	41	83	1	62	81	32	68	78	11	30	55	60	35	37	6	262	38	52	35	37	30	262
SF	13	17	13	50	1	32	55	28	27	40	5	16	12	37	24	12	1	38	112	37	24	21	21	112
SI	18	22	17	64	1	51	58	30	56	63	6	23	20	46	30	25	4	52	37	161	30	20	21	161
SR	14	11	13	43	2	34	59	22	34	93	7	11	22	277	277	17	7	35	24	30	277	15	57	277
TO	19	11	22	44	2	27	71	19	31	33	9	26	28	31	15	11	4	37	21	20	15	333	14	333
US	5	7	6	36	0	20	40	13	22	49	4	10	12	68	57	12	2	30	21	21	57	14	129	129
	94	83	147	258	349	174	697	113	227	411	60	177	196	451	277	129	249	262	112	161	277	333	129	

Notes:

- Diagonal values indicate the total number of words in the entity word list (all should match)
- Table is symmetrical along diagonal i.e. can be transposed.

As can be expected, the overall overlap between models is relatively low. Although very large models such as CYC include the exact same terminology as some of the other smaller ones, there is a major methodological problem. Many models may use synonyms instead of identical words to indicate the same concept. Indeed, [HONK98] reports that the chance of two (experienced) modellers using the same term for a given domain concept is less than 20%.

Hence a more sophisticated approach was adopted in order to capture a match between concepts that have the same meaning but use different words: the problem of synonyms. One possible automated approach for trying to cope with synonyms can be found in [AGIR00] who uses topic signatures based on extensive analysis of web pages returned by search engines. However, this approach is computationally not feasible for large vocabulary sets. Another approach, suggested in [NOY00] for the purpose of merging ontologies, requires significant input by a human expert.

Fortunately, there is a computationally more tractable and methodologically much more defensible way to deal with synonyms: WordNet [SWAR96]. WordNet is "an online lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets." [http://www.cogsci.princeton.edu/~wn/]. This database and associated software was developed by the Cognitive Science Laboratory at Princeton University and has been made available for linguistic and related research. Many articles have been written about WordNet and it is widely used in the linguistics research community.

An extra feature of WordNet is that it also includes other lexical relationships, including hypernyms: more general terms which are the equivalent of super-classes. This enables specializations to be mapped against generalizations, even if the modelling language does not allow generalizations, and these concepts were therefore excluded from the model.

For this research, WordNet 1.6 was used to generate synonyms and hypernyms for each word in each model list. As an example, the following would be returned by WordNet when requested for noun synonyms for the (AIAI entity) word "action".

```
Noun synonyms generated by WordNet 1.6 for the AIAI model word "action".

Synonyms/Hypernyms (Ordered by Frequency) of noun action
9 senses of action
Sense 1
action
       => act, human action, human activity
Sense 2
action, activity, activeness
       => state
Sense 3
legal action, action, action at law
       => proceeding, legal proceeding, judicial proceeding, proceedings
Sense 4
military action, action
       => group action
Sense 5
action
       => group action
Sense 6
natural process, natural action, action, activity
       => process
Sense 7
```

```
action
       => plot
Sense 8
action
       => mechanism
Sense 9
action
       => drive
```

The following procedure was followed to create word lists with synonyms for each of the model entities for each of the models.

1. The word list containing the parsed model entity words ML, with the duplicates removed, was selected.

2. Each of these model words was submitted to the WordNet v. 1.6 *nouns* database. For simplicity but also because "proper entities" can be expected to be nouns, no attempt was made to check the verbs, adjectives or adverbs databases.

3. For each of the model words, the list with synonyms and immediate hypernyms for the *first two most frequently used* noun senses was requested. Retrieving data for less frequently used word senses was expected to result in obscure or non-intended synonyms. For example, the two most frequently used senses of the model word "action" are "human action" (as in "we need to take immediate action to prevent a re-occurrence of this event"); and "state of activity" (as in "let's see some action here!"). It is unlikely that the (AIAI) model builders specifically intended "action" in the sense of "plot" or "mechanism".

4. Only the first 10 synonyms/hypernyms were considered for either of the word senses (very few words had more than 10 synonyms for any one word sense). Note however, that additional synonyms are also given as part of the "sense description". For instance, in order to define or describe the second sense of "action", WordNet first returns "activity" and "activeness" as part of its synset definition and then suggests the synonym "state". So, theoretically a maximum of 40 synonyms was possible for each term.

5. Synonyms consisting of multiple words were deleted. Unlike the case of perspicuity where meaning was subordinate to word form, it was considered that if these would have been parsed, a significant change in meaning could result. For the first sense of "action", both the synonyms "human action" and "human activity" would have been removed, because the term "human" is not a synonym of "action". Although the terms "action" and "activity" are therefore also deleted (and lost), they reappear as stand-alone (single-word) synonyms for the second sense of the word.

6. Duplicate synonyms were removed (the same synonym may exist for the same word in different word senses see e.g. the synonym "group action" for senses 4 and 5 of the model word "action").

A new list of words for the model, EML (*"Extended ML"*), could now be compiled which included frequently used synonyms and hypernyms for all the original terms in ML. The EML was sorted alphabetically for computational efficiency.

The procedure of matching words was repeated but now entity words in the original model word list ML (in model A) were now matched against the words in the extended word list EML containing synonyms and hypernyms in model B. Note that that **no attempt was made**

to map the synonym for an entity against a synonym of the other list because that could create spurious or even incorrect mappings. In other words, ML_A was mapped against EML_B, not EML_A against EML_B!

When using synonyms for the lookup, a new problem is introduced: now two different original entities (words) from the one model can correspond to one single entity in the other model. For example, the two different concepts of allocation and share could be mapped to the single concept of allotment in another model (which has allocation and share as synonyms). This distorts the symmetry of the overlap calculations in that the number of concepts from one model X found in another model Y, may be different to the number of concepts from model Y found in model X. In these cases, the "model overlap" was calculated as the average of the two values. There should be no major implication for the overall validity since the numbers involved are relatively small. The original table with raw match counts before averaging is found in Appendix K.

Note that future research could refine the concept of the "binary" matching of synonyms (whether or not the entity from one model is identical to the synonym of an entity of another model) with a "fuzzy" match which can take on a value ranging between a full to no match. A first possible more refined measure is the concept of "semantic relatedness" i.e. measuring the semantic distance between two entities. Unfortunately, in order to follow this approach, there are some methodological issues to be resolved including the choice of similarity measure ([BUDA01] mentions such measures which could be calculated using WordNet but there appears to be no sound theoretical ground for choosing any one of them) as well as whether the distance should be calculated between an entity and its closest equivalent in the other model (if any?) or to all other entities in the target model. Since this line of approach introduces more problems than it solves, it was not pursued any further. Where models are very similar, a case can be made to revisit this approach.

A second, very different, approach to measure concept similarity would be to use information present in the model itself to construct a non-binary similarity measure for matching concepts. The advantage of these methods is that they do not rely on an external reference. The most feasible approach to this type of distance modelling is based on the number of attributes that the corresponding entities from different models have in common, as suggested in [BISS00] and [MAED01a]. Because few models in the database include attributes, this approach could not be applied in this research either. An alternative approach is to use the concept of "relational similarity" as proposed by [MONT00] whereby the degree of connection between pairs of similar concepts is quantified by taking into account the relative connectedness of the concepts. Their approach can be modified to apply to quantify the similarity of individual entities between two models (using their fan-outs) but the underlying methodological rationale for doing this then falls away.

Notwithstanding the above methodological options, Table 8-14 and Table 8-15 detail the result of the mapping using a binary similarity index. Table 8-14 lists absolute counts of matched terms, Table 8-15 expresses the overlaps as percentages (fractions) of the original list (i.e. ML_A).

Table 8-14: Number of Matches Between the Model Entities of One Model With the List of Synonyms For the Model Entities of the Other Model.

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US	
AI	94	20	29	46	2	36	78	32	48	40	10	33	32	46	31	19	15	48	26	33	31	36	16	94
AK	20	83	22	42	4	33	72	25	41	55	12	26	25	38	25	18	15	45	31	35	25	22	19	83
AR	29	22	147	59	2	46	78	28	54	57	18	29	61	54	33	43	15	68	26	37	33	39	15	147
BA	46	42	59	258	6	93	164	69	108	130	24	61	83	121	84	62	37	124	71	90	84	72	57	258
BE	2	4	2	6	349	4	7	3	7	8	2	1	4	7	5	3	2	4	3	4	5	3	1	349
BO	36	33	46	93	4	174	129	50	91	96	20	40	59	90	67	46	22	98	51	74	67	52	41	174
CY	78	72	78	164	7	129	697	93	137	185	38	91	99	178	125	77	61	162	97	111	125	131	76	697
FO	32	25	28	69	3	50	93	113	67	61	9	50	38	57	41	31	21	63	40	52	41	39	25	113
HA	48	41	54	108	7	91	137	67	227	117	22	65	77	105	74	65	33	115	54	91	74	64	46	227
IN	40	55	57	130	8	96	185	61	117	411	25	54	93	180	133	75	39	121	67	103	133	67	79	411
MI	10	12	18	24	2	20	38	9	22	25	60	13	16	22	13	12	7	27	16	18	13	19	11	60
NH	33	26	29	61	1	40	91	50	65	54	13	177	36	51	32	29	21	64	36	47	32	50	23	177
NI	32	25	61	83	4	59	99	38	77	93	16	36	196	80	49	62	21	94	34	55	49	51	31	196
OB	46	38	54	121	7	90	178	57	105	180	22	51	80	451	291	60	35	110	62	90	291	68	95	451
OD	31	25	33	84	5	67	125	41	74	133	13	32	49	291	277	40	23	73	43	65	277	37	78	277
PU	19	18	43	62	3	46	77	31	65	75	12	29	62	60	40	129	11	70	30	45	40	27	27	129
RA	15	15	15	37	2	22	61	21	33	39	7	21	21	35	23	11	249	31	18	24	23	21	15	249
SA	48	45	68	124	4	98	162	63	115	121	27	64	94	110	73	70	31	262	66	92	73	68	48	262
SF	26	31	26	71	3	51	97	40	54	67	16	36	34	62	43	30	18	66	112	56	43	39	36	112
SI	33	35	37	90	4	74	111	52	91	103	18	47	55	90	65	45	24	92	56	161	65	45	43	161
SR	31	25	33	84	5	67	125	41	74	133	13	32	49	291	277	40	23	73	43	65	277	37	78	277
TO	36	22	39	72	3	52	131	39	64	67	19	50	51	68	37	27	21	68	39	45	37	333	32	333
US	16	19	15	57	1	41	76	25	46	79	11	23	31	95	78	27	15	48	36	43	78	32	129	129
	94	83	147	258	349	174	697	113	227	411	60	177	196	451	277	129	249	262	112	161	277	333	129	

Notes

- Table has been made diagonally symmetrical by averaging each cell with its corresponding value in the transposed table (see note above).
- It is possible for extremely close models to have more matches than the smaller model has entity words e.g. when OB is mapped against OD, some words in OB are specializations of words in OD. Averaging reduces this impact.

Table 8-15: Synonym-based Matches Expressed as a Percentage of the Base Model i.e. Relative Overlap as Percentage of the Model.

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US
AI		21%	30%	49%	2%	38%	83%	34%	51%	42%	11%	35%	34%	49%	32%	20%	15%	51%	28%	35%	32%	38%	16%
AK	24%		26%	50%	4%	40%	86%	30%	49%	66%	14%	31%	30%	46%	30%	22%	17%	54%	37%	42%	30%	27%	22%
AR	19%	15%		40%	1%	31%	53%	19%	37%	39%	12%	19%	41%	36%	22%	29%	10%	46%	18%	25%	22%	27%	10%
BA	18%	16%	23%		2%	36%	63%	27%	42%	50%	9%	23%	32%	47%	33%	24%	14%	48%	28%	35%	33%	28%	22%
BE	0%	1%	0%	2%		1%	2%	1%	2%	2%	0%	0%	1%	2%	1%	1%	1%	1%	1%	1%	1%	1%	0%
BO	21%	19%	26%	53%	2%		74%	28%	52%	55%	11%	23%	34%	51%	39%	26%	12%	56%	29%	43%	39%	30%	24%
CY	11%	10%	11%	23%	1%	19%		13%	20%	26%	5%	13%	14%	25%	18%	11%	9%	23%	14%	16%	18%	19%	11%
FO	28%	22%	25%	61%	2%	44%	82%		59%	54%	8%	44%	33%	50%	36%	27%	18%	55%	35%	46%	36%	34%	22%
HA	21%	18%	24%	48%	3%	40%	60%	29%		52%	10%	28%	34%	46%	33%	28%	14%	50%	24%	40%	33%	28%	20%
IN	10%	13%	14%	32%	2%	23%	45%	15%	28%		6%	13%	23%	44%	32%	18%	9%	29%	16%	25%	32%	16%	19%
MI	17%	19%	29%	39%	3%	33%	63%	14%	37%	41%		21%	26%	36%	21%	20%	11%	44%	27%	29%	21%	31%	18%
NH	19%	14%	16%	34%	1%	23%	51%	28%	36%	31%	7%		20%	29%	18%	16%	12%	36%	20%	26%	18%	28%	13%
NI	16%	13%	31%	42%	2%	30%	51%	19%	39%	47%	8%	18%		41%	25%	32%	10%	48%	17%	28%	25%	26%	16%
OB	10%	8%	12%	27%	1%	20%	39%	13%	23%	40%	5%	11%	18%		65%	13%	8%	24%	14%	20%	65%	15%	21%
OD	11%	9%	12%	30%	2%	24%	45%	15%	27%	48%	5%	12%	18%	105%		14%	8%	26%	15%	23%	100%	13%	28%
PU	14%	14%	33%	48%	2%	36%	59%	24%	50%	58%	9%	22%	48%	47%	31%		9%	54%	23%	35%	31%	21%	21%
RA	6%	6%	6%	15%	1%	9%	24%	8%	13%	15%	3%	8%	8%	14%	9%	4%		12%	7%	9%	9%	8%	6%
SA	18%	17%	26%	47%	2%	37%	62%	24%	44%	46%	10%	24%	36%	42%	28%	27%	12%		25%	35%	28%	26%	18%
SF	23%	27%	23%	63%	2%	45%	86%	36%	48%	60%	14%	32%	30%	55%	38%	27%	16%	58%		50%	38%	35%	32%
SI	20%	22%	23%	56%	2%	46%	69%	32%	56%	64%	11%	29%	34%	56%	40%	28%	15%	57%	34%		40%	28%	26%
SR	11%	9%	12%	30%	2%	24%	45%	15%	27%	48%	5%	12%	18%	105%	100%	14%	8%	26%	15%	23%		13%	28%
TO	11%	7%	12%	21%	1%	16%	39%	12%	19%	20%	6%	15%	15%	20%	11%	8%	6%	20%	12%	14%	11%		9%
US	12%	14%	11%	44%	1%	32%	59%	19%	36%	61%	8%	17%	24%	73%	60%	21%	11%	37%	28%	33%	60%	24%	

Note: this table is not symmetrical because the base lexicon usually has a different number of words than the target lexicon.

These tables show a marked increase in the number of mappings (correspondence) between models. For instance, the number of concepts mapped literally between Baan and SAP was 83 i.e. 83 entity words were the same in Baan and SAP, representing 32% of the entity words of SAP or Baan (they have almost the same number of entity words). When synonyms are taken into account, 124 entities can be mapped, representing 47% of the words!

For smaller models, the improvement is often even more dramatic. For example, there are only 8 common words between the AKMA and AIAI models. With synonyms, this increases to 20 words i.e. an increase from 10% to 24% for AKMA and from 9% to 21% for AIAI. For the entire table, the average overlap (excluding the degenerate 100% overlaps on the diagonal, between a model and itself) increases from 29 to 53 concept.! In relative terms, using synonyms improves the average overlap between models from 14% to 26% - almost a doubling!

It is possible to use the relative percentages to see which models are closer to each other than others. The fact that models are of different size complicates interpretation. How does the 10% overlap of the large CYC model to the AKMA model compare with the 9% overlap of the Baan model with the much smaller Miller model? The short answer is: it is impossible to compare the two figures.

Another problem with the tables is that the analysis is hampered due to the large volume of data (23 by 23 = 529 cells!). In order to make the data more accessible, it is necessary to pursue the similarity analysis by means of more statistical methods.

8.7.2 Calculating the Similarity Matrices.

A preferred approach to comparing the degree of overlap is by using statistical similarity measures. Statisticians have designed a fairly large number of similarity measures, e.g. Statistica computes the following measures: matching, Jaccard, Russell & Rao, Hamman, dice, antiDice, Sneath & Sokal, Rogers & Tanimoto, Ochiai, Yule, Anderberg, Kulczynski, Gower2 and Pearson distances. Many of these have not been validated in a linguistic context. In addition, some of them cannot be used in the case where vectors are of unequal length (the word lists for each model are different sizes). After consulting a substantial body of literature including [ALJL01; BISS00; CHEN00; CHEN97; DAST97; DUCH00; ISAA99; LEE97; LIN98; MAED01a], the following three measures appeared to be most commonly used in linguistics analysis: the cosine, dice, and Jaccard distance. There does not appear to be any convincing conceptual argument about which measure gives the best results, but an empirical study in linguistic analysis suggested that the Jaccard similarity measure may yield better predictive matching results than the cosine or dice measure [IBRA02a; IBRA02b]. One can calculate either a distance measure, whereby 1 (or 100%) means a far away as possible and 0 means identity; or the complementary similarity measure, which is equal to 1 minus the distance coefficient i.e. 1 = maximal similarity and 0 = no similarity. Similarity measures are used in the tables for this research.

Table 8-16: Dice Similarity Coefficients for Models Based on Synonyms.

Model	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US
AI	100%	23%	24%	26%	1%	27%	20%	30%	30%	16%	13%	24%	22%	17%	16%	17%	8%	27%	25%	25%	16%	17%	14%
AK	23%	100%	19%	24%	2%	26%	18%	25%	26%	22%	16%	20%	18%	14%	14%	17%	9%	26%	31%	29%	14%	11%	17%
AR	24%	19%	100%	29%	1%	29%	18%	22%	29%	20%	17%	18%	35%	18%	15%	31%	7%	33%	20%	24%	15%	16%	11%
BA	26%	24%	29%	100%	2%	43%	34%	37%	45%	39%	15%	28%	36%	34%	31%	32%	14%	48%	38%	43%	31%	24%	29%
BE	1%	2%	1%	2%	100%	2%	1%	1%	2%	2%	1%	0%	1%	2%	1%	1%	1%	1%	1%	1%	1%	1%	0%
BO	27%	26%	29%	43%	2%	100%	30%	34%	45%	33%	17%	23%	32%	29%	30%	30%	10%	45%	35%	44%	30%	21%	27%
CY	20%	18%	18%	34%	1%	30%	100%	23%	30%	33%	10%	21%	22%	31%	26%	19%	13%	34%	24%	26%	26%	25%	18%
FO	30%	25%	22%	37%	1%	34%	23%	100%	39%	23%	10%	34%	24%	20%	21%	25%	11%	33%	36%	38%	21%	17%	21%
HA	30%	26%	29%	45%	2%	45%	30%	39%	100%	37%	15%	32%	36%	31%	29%	36%	14%	47%	32%	47%	29%	23%	26%
IN	16%	22%	20%	39%	2%	33%	33%	23%	37%	100%	10%	18%	30%	42%	39%	28%	12%	36%	26%	36%	39%	18%	29%
MI	13%	16%	17%	15%	1%	17%	10%	10%	15%	10%	100%	11%	12%	8%	7%	13%	4%	16%	19%	16%	7%	9%	11%
NH	24%	20%	18%	28%	0%	23%	21%	34%	32%	18%	11%	100%	19%	16%	14%	19%	10%	29%	25%	28%	14%	19%	15%
NI	22%	18%	35%	36%	1%	32%	22%	24%	36%	30%	12%	19%	100%	25%	21%	38%	9%	41%	22%	31%	21%	19%	19%
OB	17%	14%	18%	34%	2%	29%	31%	20%	31%	42%	8%	16%	25%	100%	80%	21%	10%	31%	22%	29%	80%	17%	33%
OD	16%	14%	15%	31%	1%	30%	26%	21%	29%	39%	7%	14%	21%	80%	100%	20%	9%	27%	22%	30%	100%	12%	38%
PU	17%	17%	31%	32%	1%	30%	19%	25%	36%	28%	13%	19%	38%	21%	20%	100%	6%	36%	25%	31%	20%	12%	21%
RA	8%	9%	7%	14%	1%	10%	13%	11%	14%	12%	4%	10%	9%	10%	9%	6%	100%	12%	10%	11%	9%	7%	8%
SA	27%	26%	33%	48%	1%	45%	34%	33%	47%	36%	16%	29%	41%	31%	27%	36%	12%	100%	35%	43%	27%	23%	25%
SF	25%	31%	20%	38%	1%	35%	24%	36%	32%	26%	19%	25%	22%	22%	22%	25%	10%	35%	100%	41%	22%	18%	30%
SI	25%	29%	24%	43%	1%	44%	26%	38%	47%	36%	16%	28%	31%	29%	30%	31%	11%	43%	41%	100%	30%	18%	29%
SR	16%	14%	15%	31%	1%	30%	26%	21%	29%	39%	7%	14%	21%	80%	100%	20%	9%	27%	22%	30%	100%	12%	38%
TO	17%	11%	16%	24%	1%	21%	25%	17%	23%	18%	9%	19%	19%	17%	12%	12%	7%	23%	18%	18%	12%	100%	14%
US	14%	17%	11%	29%	0%	27%	18%	21%	26%	29%	11%	15%	19%	33%	38%	21%	8%	25%	30%	29%	38%	14%	100%

Table 8-17: Ranking the Similarity Between Models Using Synonyms (1 = most similar, 264 = least similar).

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US
AI		139	127	90	249	101	80	77	59	157	201	115	132	141	170	184	223	82	118	112	170	154	198
AK	139		165	98	233	103	87	119	86	93	187	147	162	161	185	183	219	83	66	81	185	202	177
AR	127	165		84	251	95	128	142	89	134	174	180	42	151	188	71	228	47	155	131	188	181	214
BA	90	98	84		234	13	28	21	9	22	172	99	37	44	67	52	196	4	15	12	67	126	69
BE	249	233	251	234		237	242	245	232	235	247	253	238	236	240	246	250	243	244	239	240	248	252
BO	101	103	95	13	237		35	45	8	41	163	138	64	63	78	75	216	7	40	11	78	143	104
CY	80	87	128	28	242	35		55	50	49	175	114	111	65	96	117	193	33	48	56	96	108	122
FO	77	119	142	21	245	45	55		16	100	217	46	121	123	135	120	206	39	43	31	135	160	152
HA	59	86	89	9	232	8	50	16		29	173	61	38	57	91	34	199	6	53	5	91	133	110
IN	157	93	134	22	235	41	49	100	29		190	158	58	14	24	60	210	36	70	23	24	176	51
MI	201	187	174	172	247	163	175	217	173	190		209	197	204	221	200	231	146	164	179	221	203	212
NH	115	147	180	99	253	138	114	46	61	158	209		169	178	194	167	220	88	116	102	194	153	192
NI	132	162	42	37	238	64	111	121	38	58	197	169		109	148	27	224	17	140	76	148	159	168
OB	141	161	151	44	236	63	65	123	57	14	204	178	109		2	125	218	62	105	54	2	182	26
OD	170	185	188	67	240	78	96	135	91	24	221	194	148	2		144	225	106	129	73	1	207	19
PU	184	183	71	52	246	75	117	120	34	60	200	167	27	125	144		230	32	124	72	144	205	150
RA	223	219	228	196	250	216	193	206	199	210	231	220	224	218	225	230		211	215	213	225	229	227
SA	82	83	47	4	243	7	33	39	6	36	146	88	17	62	106	32	211		30	10	106	137	113
SF	118	66	155	15	244	40	48	43	53	70	164	116	140	105	129	124	215	30		18	129	156	85
SI	112	81	131	12	239	11	56	31	5	23	179	102	76	54	73	72	213	10	18		73	166	94
SR	170	185	188	67	240	78	96	135	91	24	221	194	148	2	1	144	225	106	129	73		207	19
TO	154	202	181	126	248	143	108	160	133	176	203	153	159	182	207	205	229	137	156	166	207		191
US	198	177	214	69	252	104	122	152	110	51	212	192	168	26	19	150	227	113	85	94	19	191	

Table 8-16 gives the Jaccard similarity for the various models. The tables for the cosine and the dice distances are included in Appendix K and the similarity formulae in Appendix J Note that the similarity measures used are the ones suggested by [DUCH00] i.e. adjusted for unequal vector sizes. Unlike the relative overlap measure calculated in the previous table, all three indices are symmetrical i.e. the similarity from model A to model B is the same as from model B to A, regardless of their relative sizes.

The following important methodological notes are in order:

- The Jaccard similarity coefficients yield fairly low numbers – the cosine and dice coefficients are generally almost twice as high (see Appendix K). The cosine and dice coefficients are actually very close to the "relative overlap" percentages.
- Due to the way the Jaccard and dice coefficients are calculated, they preserve relative ranking almost perfectly i.e. if the Jaccard coefficient for two models exceeds that for two other models, then the dice coefficient will almost certainly also be larger. Hence the ranking tables for Jaccard and dice similarity coefficients are virtually the same, despite the fact that the actual coefficients are quite different.

8.7.3 Analyzing the Similarity Coefficients using Ranking and Hierarchical Tree Analysis.

First analysis is the most similar and most dissimilar models. From Table 8-16 with the similarity coefficients, Table 8-17 was derived by ranking each similarity coefficient from 1 (closest or highest similarity) to 253 (lowest similarity). Note that the degenerate values of perfect similarity for the diagonal cells (each model is 100% similar to itself) were deleted.

Table 8-18 lists the most and least similar models.

Table 8-18: Most and Least Similar Model Pairs.

Rank	Most similar	Rank	Least similar	Rank	Least similar
1	SR - OD	214	US - AR	234	BE - BA
2	OB - SR	215	SF - RA	235	IN - BE
2	OB - OD	216	RA - BO	236	OB - BE
4	SA - BA	217	MI - FO	237	BO - BE
5	SI - HA	218	RA - OB	238	NI - BE
6	SA - HA	219	RA - AK	239	SI - BE
7	SA - BO	220	RA - NH	240	BE - SR
8	HA - BO	221	MI - SR	240	OB - BE
9	HA - BA	221	OB - MI	242	CY - BE
10	SI - SA	223	RA - AI	243	SA - BE
11	SI - BO	224	RA - NI	244	SF - BE
12	SI - BA	225	RA - OD	245	FO - BE
13	BO - BA	225	SR - RA	246	PU - BE
14	OB - IN	227	US - RA	247	MI - BE
15	SF - BA	228	RA - AR	248	TO - BE
16	HA - FO	229	TO - RA	249	BE - AI
17	SA - NI	230	RA - PU	250	RA - BE
18	SI - SF	231	RA - MI	251	BE - AR
19	SR - US	232	HA - BE	252	US - BE
19	US - OB	233	BE - AK	253	NH - BE

The following observations can be made:

- Most similar, in fact displaying perfect similarity, are SR and OD. This should be the case since SemiRandom was constructed using exactly the same entity list. Because of this, the duplicate rankings occur throughout the table because the similarity/distance from any model to SR is the same as that to OD.
- Second most similar is OB to SR (and OD). Again, this should not surprise us, since OD is a sub-set of OB (all the entities with 2 or more relationships). This is also the result of the (artificial) construction of SR and OD.
- The next two most similar models are the SAP and the Baan model. Both are ERP models of roughly the same size and functionality and this is the strongest possible confirmation of the validity of the technique or methodological procedure that has been adopted.
- Almost equally similar, with rank 5, are the Hay and Silverston models. Again, this validates the measure since both models come from the same reference discipline (data model libraries) and were published in the same year.
- The next most similar models are SAP and Hay, with SAP and BOMA following very close behind.
- From there, there are a number of different combinations between the above, which all seem to form a fairly close or similar cluster of models.
- The first "new" set of similar models are OB and Inmon, ranked 14th, which is perhaps somewhat of a surprise until one realizes that the Inmon model, designed for data warehousing, was more of a list of terms and thus close to a dictionary approach, than a formal or proper model.
- At the bottom end, there are two or three models that are completely unlike any of the other models.
- The furthest removed is the BelgAcc model, not because of the intrinsically different domain or reference discipline, but because the language used (Dutch) is different to the one adopted for the other models (English). A very few words are the same in both languages (e.g. "product") but there all similarity ends.
- Next, again in a category of its own, is the Random model, whose domain was a randomly selected set of words from the Oxford Paperback Dictionary. Again, this validates the technique.
- Finally, the third model that cannot be partnered with any other model is the Miller model, which comes from a very different reference discipline and has a very peculiar choice of entities.

Another type of analysis is to investigate for *each model* which its closest or most similar neighbours are. This will again depend on the choice of similarity measure used and Table 8-19 lists the closest three neighbours for each model using dice and cosine similarity. Using the Jaccard similarity measure adds no new information since it replicates the dice table almost perfectly, and has therefore been omitted.

Table 8-19: Three Most Similar Neighbours for Each Model.

	DICE similarity-based closest neighbours							COSINE similarity-based closest neighbours					
	Closest		2nd close		3rd close			Closest		2nd close		3rd close	
Model	1	Sim	2	Sim	3	Sim	Model	1	Sim	2	Sim	3	Sim
AI	FO	30%	HA	30%	BO	27%	AI	HA	33%	FO	31%	CY	30%
AK	SF	31%	SI	29%	HA	26%	AK	SF	32%	SI	30%	SA	30%
AR	NI	35%	SA	33%	PU	31%	AR	NI	36%	SA	35%	PU	31%
BA	SA	48%	HA	45%	SI	43%	BA	SA	48%	HA	45%	SI	44%
BE	HA	2%	BA	2%	IN	2%	BE	HA	2%	AK	2%	BA	2%
BO	HA	45%	SA	45%	SI	44%	BO	SA	46%	HA	46%	SI	44%
CY	BA	34%	SA	34%	IN	33%	CY	BA	39%	SA	38%	BO	37%
FO	HA	39%	SI	38%	BA	37%	FO	HA	42%	BA	40%	SI	38%
HA	SA	47%	SI	47%	BO	45%	HA	SI	47%	SA	47%	BO	46%
IN.	OB	42%	BA	39%	OD	39%	IN	OB	42%	BA	40%	SI	40%
MI	SF	19%	BO	17%	AR	17%	MI	SA	21%	BO	20%	SF	20%
NH	FO	34%	HA	32%	SA	29%	NH	FO	35%	HA	32%	SA	30%
NI	SA	41%	PU	38%	HA	36%	NI	SA	41%	PU	39%	BA	37%
OB	OD	80%	SR	80%	IN	42%	OB	OD	82%	SR	82%	IN	42%
OD	SR	100%	OB	80%	IN	39%	OD	SR	100%	OB	82%	US	41%
PU	NI	38%	HA	36%	SA	36%	PU	NI	39%	SA	38%	HA	38%
RA	BA	14%	HA	14%	CY	13%	RA	CY	15%	BA	14%	HA	14%
SA	BA	48%	HA	47%	BO	45%	SA	BA	48%	HA	47%	BO	46%
SF	SI	41%	BA	38%	FO	36%	SF	BA	42%	SI	41%	SA	38%
SI	HA	47%	BO	44%	SA	43%	SI	HA	47%	SA	45%	BO	44%
SR	OD	100%	OB	80%	IN	39%	SR	OD	100%	OB	82%	US	41%
TO	CY	25%	BA	24%	HA	23%	TO	CY	27%	BA	24%	HA	23%
US	OD	38%	SR	38%	OB	33%	US	OD	41%	SR	41%	OB	39%

Table 8-19 confirms much of the top and bottom ranking analysis above:

- The very close similarity between the model pairs OB-SR; OD-SR; SAP-Baan; Silverston-Hay; Inmon-OB.
- The three models that are very dissimilar to other models, Random, BelgAcc and Miller. Their dice similarity coefficient to the nearest neighbours is less than 20%.

Some interesting additional observations emerge from the above analysis:

- The closeness between ARRI and Nippon. It will be recalled that both models are CIM/Enterprise Engineering inspired.
- Similarly, the model closest to the Purdue model is the Nippon model. It will be remembered that the Nippon model appeared in the appendix of the Purdue model documentation!
- Fairly dissimilar but still the closest neighbour to the TOVE model is CYC, also from the same reference discipline: ontology research.
- Surprisingly, the AIAI model is closer to the data models (Fowler, Hay, BOMA) than any other model.

Finally, it is interesting – and heartening – to note that the choice of similarity measure does not really affect the analysis. Although there are some differences in the rankings, all of the above remarks are valid regardless of whether the cosine, the dice or the Jaccard coefficients are used.

The final analysis extends and confirms the above analysis more systematically by employing the most appropriate tool from the cluster analysis arsenal: hierarchal tree construction. Because the study described in [IBRA02a and IBRA02b] suggests that the Jaccard similarity coefficient is empirically the best, it will be used for the remainder of the discussion. Although it was argued above that the dice coefficient gives a better indication of the actual amount of overlap between models, the absolute value of the similarity does not play any significant role for the cluster analysis which follows.

The joining or tree clustering method uses the dissimilarity between the various models, to analyse step-by-step how a family or relatedness tree of models could be constructed. Note that the Jaccard distance is used here, which is calculated as 1 minus the Jaccard similarity coefficient.

Table 8-20 illustrates the procedure followed in constructing the hierarchical or joining tree. Note that there are various clustering algorithms available; a simple, a non-weighted clustering algorithm was used here whereby a (newly formed) cluster has the same weight as a single cluster in calculating the new distances. Although this is not normal statistical practice (in fact, the procedure had to be performed "manually" because of this), it permits a look at the relative distances between references disciplines (see below) without this being biased by the number of models in each reference discipline. For instance, when SR and OD and OB are joined, the resultant cluster should not receive a weight which is three times that of IN when it joins them later.

Table 8-20: Stepwise Distance Calculation For Similarity Dendogram.

Cluster	Model1	Model2	Distance
C1	SR	OD	0.0000000
C2	C1	OB	0.3340961
C3	SA	BA	0.6868687
C4	SI	HA	0.6957983
C5	BO	C4	0.7125049
C6	C5	C3	0.7171528
C7	IN	C2	0.7482154
C8	PU	NI	0.7642586
C9	SF	C6	0.7783659
C10	FO	C9	0.7828707
C11	AR	C8	0.8006457
C12	US	C7	0.8066753
C13	NH	C10	0.8217922
C14	AI	C13	0.8483905
C15	TO	CY	0.8549194
C16	AK	C11	0.8703951
C17	C16	C14	0.8745237
C18	C12	C15	0.8887106
C19	C18	C17	0.9012483
C20	MI	C19	0.9363804
C21	RA	C20	0.9652219
C22	BE	C21	0.9955109

- The two closest models are found by looking at the smallest distance between any two models. Naturally, these are SR and OD, having the same entities and thus a distance of 0. A new model is constructed, namely cluster "C1", and substitutes or replaces both SR and OD *viz-a-viz* the other models. Because SR and OD occupied exactly the same position in the 22-dimensional space, none of the distances needs to be adjusted.

- The two models that are now closest are, not surprisingly, OB and the new C1 (substitute for both SR and OD). Again a new "virtual model" is constructed, namely C2 which occupies a space exactly between C1 and OB. This means that all the (Jaccard) distances between the remaining models and C2 have to be recalculated.
- When the remaining 21 models are considered, it is found that the next smallest distance is between the SAP and Baan model. A substitute "cluster" model C3 is created halfway between these two and the distance between C3 and the other models is again recalculated.
- This continues until finally all models are combined in one big cluster.

This process is easy to visualize in the hierarchical or joining tree as per Figure 8-3 – this is also called a dendogram. The reader is advised to study the visualization exercise in Appendix J whereby an attempt is made to illustrate how the distance measures must be understood in multi-dimensional space for a simpler 5-dimensional similarity analysis. Most statistical programs will output a joining tree (or dendogram) with each join at a different horizontal (or vertical – depending on the orientation of the tree), but the tree has been re-drawn here to emphasize the family structure.

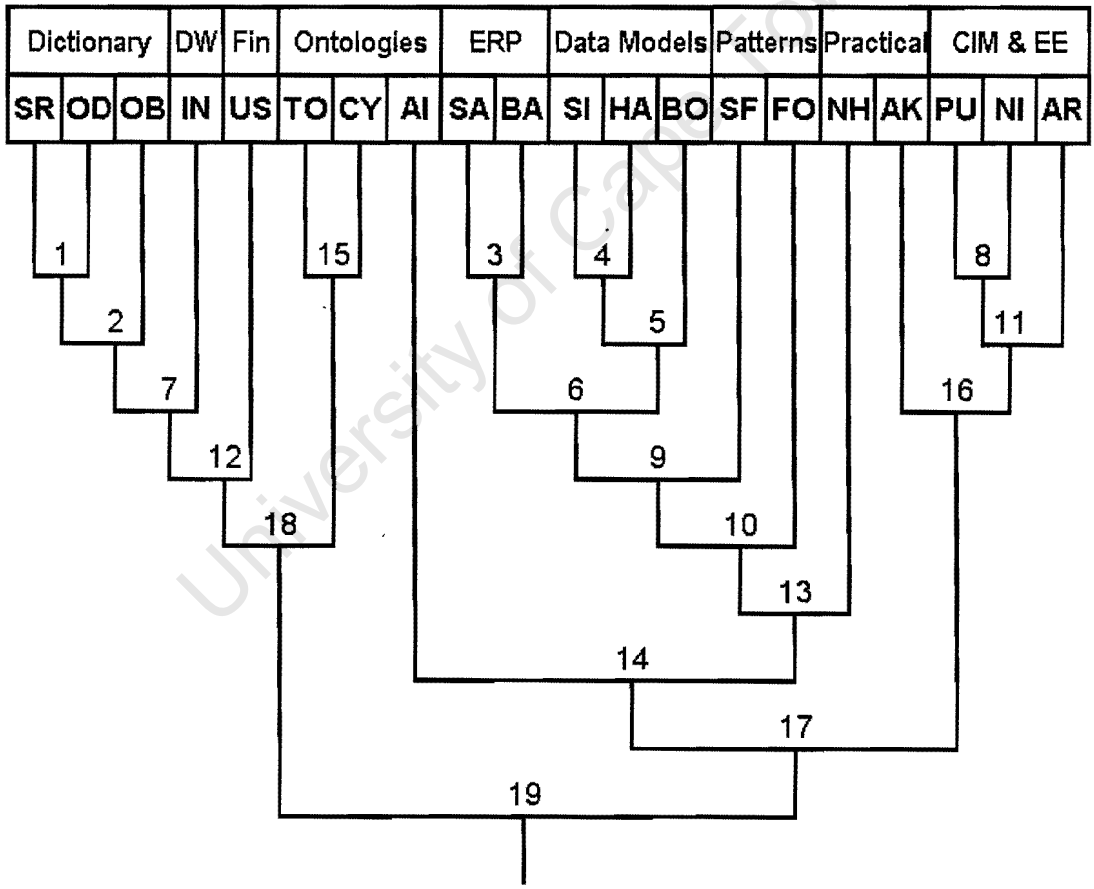


Figure 8-3: Similarity Dendogram (Hierarchical Tree).

Note how this cluster analysis confirms, and is validated by, the similarity grouping between models from the same reference disciplines. The following is an attempt to verbalize the "cluster creation" story in a way which appears to make conceptual sense.

- As explained above, the first two steps combine the models SR, OD and OB whose unusually close clustering is by virtue of their design.

- Next is the closeness of the two competing ERP systems: both are large, well-known and well-validated models, and it is not surprising to see them forming the first "real" model cluster.
- The next closest cluster (steps 4 and 5) combines the three data (library) models together: Silverston, Hay and BOMA. They follow a similar approach and cover the same domain, so it is not surprising, though a nice validation for the methodology, to see them cluster together so neatly.
- Another obvious cluster is formed by CIM/EE reference discipline based Purdue, Nippon and ARRI.
- A next step occurs where the ERP and data model clusters are joined: from a conceptual point of view, these can indeed be considered to be two disciplines that are fairly closely related.
- This ERP-Data Modelling cluster is subsequently joined by the two patterns-based models, which conceptually makes a lot of sense.
- The dictionary cluster expands by absorbing the data-warehousing Inmon model (which is indeed not much more that a vaguely structured set of terminology) as well as the USB financial model. It must be remembered that the Ottawa dictionary was also very financially oriented, as discussed when looking at perspicuity analysis.
- As seen above, the TOVE and CYC models also form their Ontology cluster, being very lexically oriented, which eventually joins the enlarged dictionary cluster.
- After this mega-cluster (C10 in the table) is joined by NHS and, somewhat surprisingly, AIAI, it joins the remaining cluster consisting of CIM & AKMA model.
- At this stage, the similarity between the clusters is fairly small. However, the two huge clusters consisting of all the models discussed so far, still have more in common than either of them shares with the "peripheral" models: Miller, Random and BelgAcc. These three are not only very far removed from the other models, but are actually even further from each other i.e. in opposite directions within the (multidimensional) space surrounding the other model clusters.

Generally, the overall shape of a dendogram tends to be either a black hole or a planetary system (see example in Appendix B). Overall, the planetary system best describes the model similarity dendogram, although the two clusters formed in stages 1 and 6 act as small black holes for a while, as they absorb their neighbours.

In an attempt to visualize the cluster analysis, Figure 8-4 shows a three-dimensional plot of the table containing the dice similarities, but with the models ordered in the same way as the hierarchical tree. The various peaks highlight clusters of similar models; the higher the peak, the more similar, the broader the peak, the more models share the similarity.

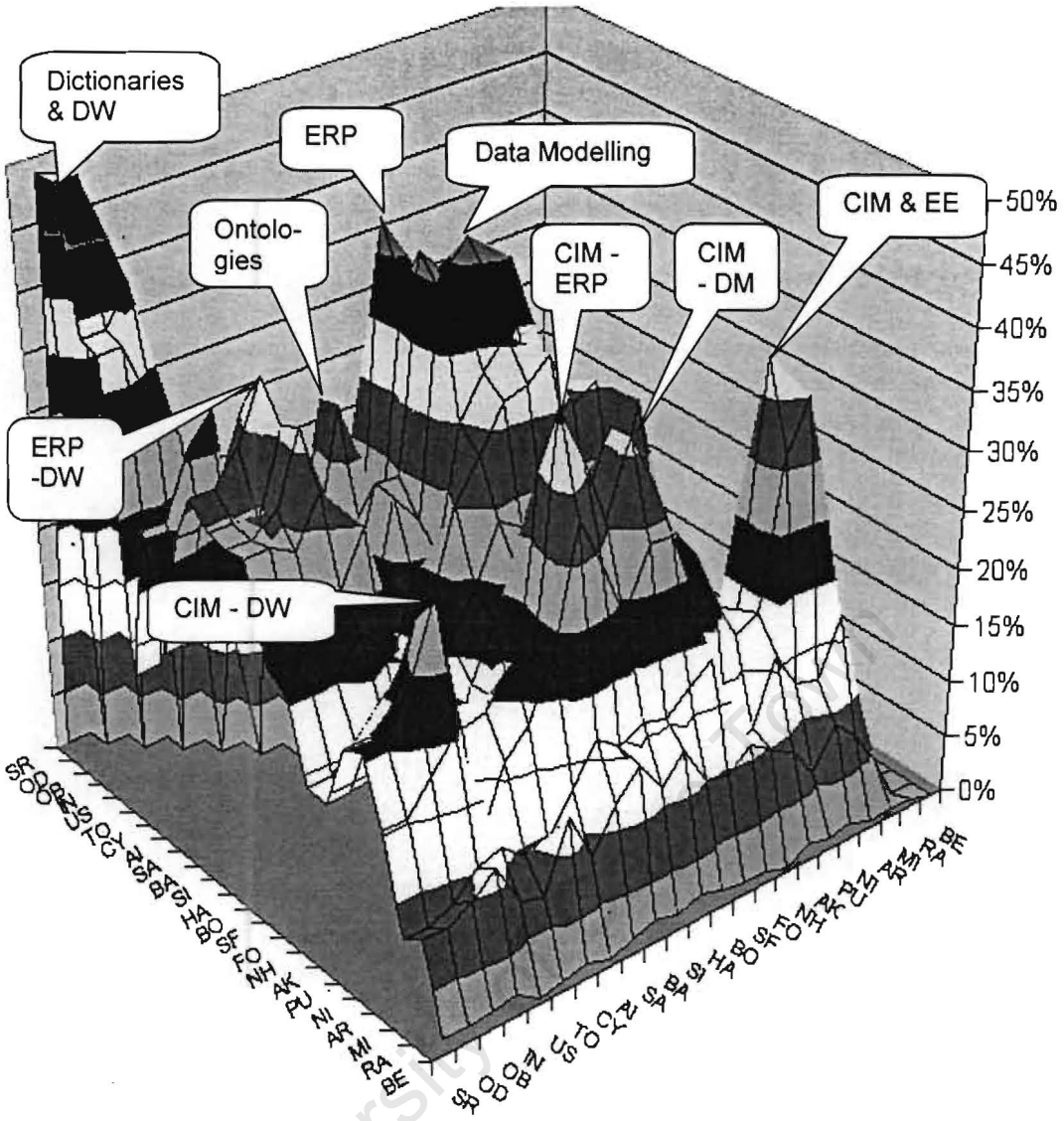


Figure 8-4: Visualization of Model Clusters.

This grouping of models according to cluster closeness confirms the validity of the clustering according to reference discipline: most of the highest peaks are located towards the back, along the diagonal line of the plot. Note that the other, symmetrical back-half of the plot has been removed to reduce clutter. The back peaks representing the dictionaries (OD, OB, SR; top truncated at 50%), ontologies and CIM & EE stand out clearly, with the most prominent and largest "triple peak" mountain at the centre back representing ERP and Data Modelling (steps 3; steps 4 & 5 combined; and step 6 in the clustering analysis are each a separate sub-peak).

However, this plot also highlights some peaks which are off the diagonal line and reveals other similarities between disciplines which were not evident in the clustering procedure (which gives an essentially one-dimension approach.) For example, although Inmon's (data warehouse) model is mathematically closest to the Ottawa model (39-42%) and part of that peak, going along its ridge from left-back to centre-front the prominent sub-peaks with the ERP and CIM/EE models are encountered. Similarly, if we follow the ridge line of the CIM/EE models (right-back to centre-front), the almost equally tall sub-peaks representing their similarity with the DM (Data Modelling), ERP and DW (Inmon) models are very prominent.

8.7.4 Most Important and Most Common Concepts

A somewhat related analysis concerns the overlap between models, not of all entities but of the most central or important entities. The importance of an entity can be calculated by means of its fan-out i.e. the number of relationships in which it participates.

For all models, the 15 entities with the highest fan-out rank were extracted. These were then compared, and any entity that appeared in more than 3 models was extracted. Table 8-21 lists the 27 concepts (or direct synonyms thereof) that featured in the top 15 concepts list for at least 3 models.

Table 8-21: Some Common Top-15 Concepts.

account	document	organization unit
activity/process/action	employee/human resource	part
address	environment	party
agent/actor	income/revenue/sales	product
asset	income statement	sale
contract/agreement	inventory/stock	supplier
corporation/business	item	time/period
cost/expense	money/cash	transaction
customer/client	order	unit

These can therefore be considered to be core concepts for any enterprise model.

A reverse analysis was done to determine how many of the above "minimally required" concepts were present in the model. When synonyms and close concepts are taken into account, most models performed well. (The synonym lists as for the "similarity analysis" above were used; but the two double-word concepts "income statement" and "organization unit" were excluded)

Table 8-22: Model Coverage of Core Concepts.

100% (all 25)	CYC
90+%	Baan, BOMA, Inmon
80+%	Hay, SAP, Ottawa-Big, Silverston
70+%	Fowler, SanFran, Purdue, TOVE
60+%	AIAI, Nippon, Ottawa-Dense & Semi-Random
50%+	AKMA, BelgAcc (translated), NHS, USB
<50%	Miller, Random.

The Miller and Random models form the exceptions – as could be expected. This analysis does not really reveal any unexpected information, and is to some extent dependent on the model size.

The methodology can be extended to include the entire model as well all meta-model elements. This is done by examining which concepts are covered by the most models i.e. the concepts which are included in the largest number of models, and to use these concepts as the basis to construct a new model which reflects the commonality (or consensus) of the models in the database. This method is explained in Chapter 10. The analysis performed above also forms a good introduction to the next type of analysis, which is concerned with the model completeness.

8.8 Model Completeness

This aspect measures how much of the domain has been covered. In principle this is easy: find an accepted description of the model domain and map each model against the concepts used. In practice there is no internationally accepted description of the "generic enterprise".

Some different approaches:

- Use the business lexicons as independent but fairly complete semantic descriptions of the domain.
- Use the Zachman framework as a conceptual guideline of which areas of business need to be covered. This approach is discussed in Chapter 10.

The approach adopted in this research will focus on the use of business lexicons to measure model completeness. Although conceptually very different, this procedure is almost the reverse of the perspicuity methodology: check each concept in the dictionary to see if the model has mapped it. Table 8-23 gives the results of how many words from each of the 5 dictionaries (as well as a combined list) could be found in the (normalized) model entity word lists.

Table 8-23: Model Completeness Measured by Coverage of Business Lexicon.

Using the original model entities							Using synonyms						
	BL	MW	OB	SB	WP	All		BL	MW	OB	SB	WP	All
AI	80	76	38	60	40	84	AI	272	201	87	146	104	287
AK	74	60	34	44	30	78	AK	233	155	68	106	74	243
AR	121	106	50	90	55	124	AR	346	254	120	204	131	352
BA	235	192	121	171	122	244	BA	636	451	221	340	249	668
BE	18	12	6	8	4	20	BE	64	51	18	28	14	77
BO	156	136	78	113	70	160	BO	452	332	161	251	162	471
CY	590	402	186	305	202	607	CY	1143	736	307	521	355	1218
FO	100	86	50	65	55	104	FO	336	233	104	160	130	349
HA	201	164	89	129	91	205	HA	574	404	182	280	201	607
IN	356	304	204	270	186	379	IN	840	616	306	463	326	914
MI	44	31	13	20	14	47	MI	168	115	47	78	53	180
NH	144	100	53	63	48	148	NH	398	259	108	157	127	414
NI	171	139	76	122	67	179	NI	507	370	170	274	182	537
OB	344	397	451	362	246	451	OB	828	721	543	562	394	983
OD	221	254	277	244	177	277	OD	573	501	369	408	297	662
PU	116	101	62	83	61	120	PU	383	282	141	211	152	398
RA	89	65	19	30	18	102	RA	439	252	83	141	108	480
SA	236	193	112	162	108	240	SA	632	449	210	319	226	673
SF	99	90	60	69	53	103	SF	310	220	109	158	125	320
SI	141	127	81	108	73	150	SI	461	327	169	259	178	486
SR	221	254	277	244	177	277	SR	573	501	369	408	297	662
TO	226	153	70	99	89	247	TO	571	369	145	238	185	621
US	113	111	92	104	91	122	US	318	259	156	209	165	344
Max	9894	3708	1063	1818	1323	11838	Max	9894	3708	1063	1818	1323	11838

Because dictionaries are likely to contain many equivalent descriptions, it was considered to be methodologically more sound to look up the words against the lists containing synonyms (developed to measure model similarity above) as well. No model comes anywhere close to covering the entire domain. The values in Table 8-23 can be converted into a relative measure by dividing each figure by the total number of words in each lexicon as reflected in the bottom row ("Max" i.e. maximum possible "score").

All percentages are fairly low:

- The highest overall coverage is, not surprisingly, the (artificially constructed) OB model which covers 451 terms (or 543 if synonyms are included) of the OB lexicon. This is equivalent to 42% (51%) of the "domain", but obviously due to the bias introduced by "constructing" the OB model as a subset of the OB lexicon, (OB consists of entities which are only those OB lexicon terms which are related to another lexicon entry). The same argument is obviously true for the SR and OD models.

- The "best" proper model is CYC which covers 29% of OB and of SB, followed by one single term by the Inmon model which also covers 29% of OB, although only 25% of SB.

The analysis is facilitated by ranking models on basis of their domain coverage, i.e. a "1" in a column indicates the model that covers most of a given lexicon (the domain); a "23" the model that maps least of a given domain.

Table 8-24: Ranking of Model Completeness Measured by Coverage of Business Lexicon.

Using the original model entities							Using synonyms						
Model	BL	MW	OB	SB	WP	All	Model	BL	MW	OB	SB	WP	All
CY	1	1	5	2	2	1	CY	1	1	4	2	2	1
OB	3	2	1	1	1	2	OB	3	2	1	1	1	2
IN	2	3	4	3	3	3	IN	2	3	5	3	3	3
OD	7	4	2	4	4	4	SA	5	7	7	7	7	4
SR	7	4	2	4	4	4	BA	4	6	6	6	6	5
TO	6	9	13	13	10	6	OD	7	4	2	4	4	6
BA	5	7	6	6	6	7	SR	7	4	2	4	4	6
SA	4	6	7	7	7	8	TO	9	10	13	12	9	8
HA	9	8	9	8	8	9	HA	6	8	8	8	8	9
NI	10	10	12	9	13	10	NI	10	9	9	9	10	10
BO	11	11	11	10	12	11	SI	11	12	10	10	11	11
SI	13	12	10	11	11	12	RA	13	17	20	20	19	12
NH	12	16	16	18	18	13	BO	12	11	11	11	13	13
AR	14	14	17	14	15	14	NH	14	14	17	18	17	14
US	16	13	8	12	8	15	PU	15	13	14	13	14	15
PU	15	15	14	15	14	16	AR	16	16	15	15	15	16
FO	17	18	17	17	15	17	FO	17	18	18	16	16	17
SF	18	17	15	16	17	18	US	18	14	12	14	12	18
RA	19	20	21	21	21	19	SF	19	19	16	17	18	19
AI	20	19	19	19	19	20	AI	20	20	19	19	20	20
AK	21	21	20	20	20	21	AK	21	21	21	21	21	21
MI	22	22	22	22	22	22	MI	22	22	22	22	22	22
BE	23	23	23	23	23	23	BE	23	23	23	23	23	23

Table 8-24 has been sorted according to the ranking for the combined lexicon list "all". The following observations can be made:

- CYC covers by far the most concepts, whichever lexicon is selected.
- Ottawa-Big does the best job of covering the domain described by the smaller lexicons MW, SB, WP and obviously the OB from which it was derived. Note that the similarity between these lexicons has already been discussed in Appendix J.
- The Inmon model ranks third for the most lexicons, although it rates second for the much larger BL, which is more different from the other models.
- OBs smaller cousins OD and SR, obviously also do a fairly good job for the smaller lexicons, although not so well for the larger BL.
- The TOVE model does a surprisingly good job of covering the BL, though it does not score well for the smaller models, probably due to its lack of coverage of financial an accounting terminology.
- Next are, finally, the ERP models where BA and SA occupy similar positions, depending on which lexicon is used. They are followed by the Nippon, BOMA and Silverston models.
- At the bottom of the table are the models with the worst coverage.

- BelgAcc finds itself at the bottom, not surprising due to its foreign language
- Miller is second-worst, not necessary due to its bad domain coverage but because it covers the domain in a very non-standard way using non-standard terminology. It is also a very small model.
- The AIAI and AKMA models are a surprise. They seem to cover very little of the domain, though their small size contributes to their bad performance.
- The Random model does not cover much of the small lexicons, although its random English words serve it well enough for the BL list to beat AIAI and AKMA.

- Other models which perform relatively badly, worse than expected perhaps, are Fowler and SanFran.
- It is noteworthy that, although the USB model does not do a good job of covering the large All or BL lexicons, it has much better domain coverage of the more financially oriented OB and SB, despite its tiny size.

As pointed out, the above analysis is obviously influenced by model size: the smaller the model, the less likely it is to cover the domain. It is possible to recalculate the above statistics in relative terms i.e. by dividing the number of matches by the size of the model. There are a number of candidates for model size in this case e.g. number of entities, or the closely correlated number of words in the normalized entity list.

When this is done (the latter size metric), the following models emerge at the top of the list.

Table 8-25: Economy of Model Coverage (Ranking).

	BL	MW	OB	SB	WP	All
FO	1	3	10	8	3	1
PU	2	1	5	1	2	2
AI	3	2	8	4	5	3
SI	4	4	6	3	6	4
MI	6	7	17	13	15	5
AK	5	10	13	14	12	6
SF	7	6	7	9	4	7
NI	9	9	11	10	11	8
BO	8	8	9	7	10	9
HA	10	13	15	16	14	10
US	11	5	3	2	1	11
BA	11	14	12	12	9	12
SA	13	16	16	17	17	13
AR	14	15	14	11	13	14
OD	16	11	1	5	7	15
SR	16	11	1	5	7	15
NH	15	19	19	19	19	17
IN	18	18	18	18	18	18
OB	19	17	4	15	16	19
RA	20	22	22	22	22	20
TO	21	20	21	21	20	21
CY	22	21	20	20	21	22
BE	23	23	23	23	23	23

Table 8-25 does *not* show how well a model covers the domain but rather how economically it does so, i.e. "wordy" or "noisy" models may include lots of terms which are not necessarily part of the domain (or the user vocabulary). This can be identified by looking at the bottom of the list:

the "guilty" models in this respect are CYC, TOVE, Random, Ottawa-Big, Inmon and NHS! On the other hand, some small models do a relatively good job of covering as much of the domain as possible, namely Fowler, Purdue, AIAI and Silverston. Note that this last analysis of relative coverage is not necessarily methodologically very sound, but may be useful in providing some additional or moderating perspectives on the absolute domain coverage metrics discussed above.

Overall, the approach of using business lexicons to measure domain coverage is computationally fairly straightforward, and appears to have some merit if synonyms are included. The measure is computationally quite involved but benefits substantially from the perspicuity analysis. However, the approach is not ideal, and it is hoped that future research may reveal a better computational way of assessing domain coverage. A possible alternative approach, using domain-related frameworks, is discussed in Chapter 10 but appears to be methodologically more flawed and substantially more subjective.

8.9 Summary and Validation of Semantic Analysis

The semantic analysis of models was concerned with the intrinsic *meaning* of the model i.e. its relationship and mapping to the underlying domain reality it is representing. Because semantic analysis is concerned with the correspondence between the abstract model and the underlying real domain, its reference disciplines are mainly linguistics, ontology research and lexicography, with much of the analysis concentrating on similarity, correspondence and cluster analysis. Many of the techniques and approaches suggested in the chapter appear not to have been used before in the area of modelling research.

Genericity can be defined narrowly as the capability of the enterprise model to be applied to enterprises in different industries, or more widely as applicability to different types of organizations. Two genericity measures were suggested and tested. A first measure focussed on the *number* of model constructs that are applicable to all domains. An alternative measure looked more at the *consistency* by which high-level constructs apply across the range of different organizations. Although the suggested methodology is not fully satisfactory and needs to be refined further, it appears to be a promising approach for future research.

The ***expressiveness*** of a model refers to the richness of the modelling language used. The key to the construction of the expressiveness score is the availability of a sufficiently detailed meta-model. The various modelling language constructs used by the model can them be mapped against the meta-model. Overall, the quantification of an expressiveness score, as the weighted index of the number of meta-model attributes covered in a model, appears to be a feasible and valid approach.

Model ***perspicuity*** and ***readability*** refer to the extent to which the model can be understood or comprehended by the intended users or readers of the model and how self-describing the model is. The perspicuity analysis was based on the matching or comparison of the model element names against common domain vocabulary lists. Although several business lexicons were investigated, the use of a well-validated corpus-based business language list annotated with word frequency statistics yielded the most valid results, especially if slightly more sophisticated wordlist preparation and matching algorithms are used.

Documentation quality can be measured fairly easily by means of a number of different qualities. ***Completeness*** looks at the proportion of model elements defined, with various possible indices depending on the representation medium used for the model (book or web). The ***inter-linkedness***

Table 8-26: Summary of Semantic Metric Validity Concerns.

Framework Criterion	Section	Metric / measurement	Recommended?	Specific validity issues (refer 4.2.3) in respect of the model sample.
Genericity	8.3	G1 Average (subjective) genericity score	N	Low construct and internal validity
		G2 (avg. score stability)	Y*	Low internal validity
Coverage	8.3	Domain coverage score	Y*	Low internal validity
Completeness	8.8	Absolute and relative lexicon coverage	N	Difficult to interpret
		Ranking of absolute lexicon coverage	Y*	A fairly high construct validity but unknown content validity
	10.1	Zachman framework mapping	N	Internal validity appears problematic
Efficiency; conciseness	8.8	Relative lexicon coverage		Construct validity appears suspect but apears to exhibit some overall criterion and content validity
Expressiveness	8.4	Avg. expressiveness score	Y	
Similarity & overlap with other models	8.7	Matrix of absolute and relative concept overlap count	N	Valid but difficult to interpret
		Matrix of (Jaccard, dice, cosine) similarity coefficients	N	Valid but difficult to interpret
		Plot of similarity coefficients	Y	Valid and visually intuitive.
		Pair-wise ranking of similarilty coefficients	N	Valid for extreme models
		Most similar neighbours	Y	Relative high construct validity: independent of similarity coefficient
		Similarity dendogram	Y	Appears to have high validity
		Most important concepts	Y	
		Core concept coverage	Y	
Perspicuity; comprehensibility; understandability; readability	8.5	GPC, WPC, RAWPC, WL-based RAWPC	N	All exhibit a high construct validity measured against each other and other lexicons
		NRAWPC	Y	Highest criterion and content validity, external validity unknown
Documentation	8.6	Completeness: OCB	Y	
		Inter-linkedness	N	Low content validity
		Extensiveness	Y	Suspect criterion validity
		Readability: FRE	Y*	Fairly high construct validity but further criterion validation suggested
		Readability: FKGL	N	Very low content validity (no discrimination)

It is again suggested that the metrics with a recommendation of "Y*" be subject to further external validity testing in future research projects.

Having completed both the syntactic and semantic model analysis, it is now appropriate to attempt the pragmatic model analysis.

Chapter 9: Pragmatic analysis

This chapter deals with the last aspect of the framework: pragmatic analysis. The previous two chapters dealt with the other two aspects of the framework, namely syntactic and semantic analysis.

Pragmatic model analysis, as defined in the framework used for this research, is concerned with those metrics and criteria which cannot be assessed purely on the basis of the information contained within the model, but which require the consideration of information regarding the use, environment or context of the model i.e. information outside the model. Hence, this chapter makes use of information not found inside the meta-model or model database. The analysis techniques falling under this heading include face validity, degree of use, authority of model author, availability, cost, flexibility, adaptability, model currency, maturity and support. Most analysis relies on the searching for and ranking of specific information details, often involving a degree of subjective interpretation and an understanding of commercial business issues.

9.1 Face Validity, Model Use and Authority

Model validity is a complex and composite measure. The various types of model validity and their interpretation in the context of enterprise modelling were discussed at the end of Chapter 5. The most succinct definition of validity, "how well the model matches reality" [WILL02], is best represented by the concept of **content validity** and, in the absence of statistical and experimental measures, is best measured by means of **face validity** i.e. the acceptance of the model by practitioners in the field.

However, model acceptance is often influenced by the authority of the developers of the model i.e. the source of the model. In the absence of contrary information, the product of a well-respected academic research group or software company will be rated higher and accepted better than that of lesser known sources. Although this is somewhat contrary to established scientific practice of "objectivity", it is a very real and pragmatic criterion used in the commercial world and cannot be ignored. Hence this is referred to as **authoritative validity**.

A possible approach to measure the face validity of a set of enterprise models would be to establish a panel of expert analysts, and request them to evaluate the various enterprise models. Apart from the cost factor, the selection of the panel would present a methodological problem since the evaluation would be influenced by the experiences and knowledge of the "experts" – especially in the light of the fact that the models originate from many different reference disciplines. Note that Bergholtz & Johannesson have used such an approach for model fragments (analysis patterns) using automated natural language-based explanation generation but their approach is not feasible for large models [BERG00].

Directly obtaining the expert opinions involves significant practical and methodological problems. Hence, a number of proxies for measuring the face validity has been investigated are suggested, using the following simplistic "stick" model which experts or organizations may use in deciding whether to adopt a given enterprise model.

(a) Information is gathered about the available enterprise models, including information about the model owner(s)/author(s).

(b) The authority of the respective model owners is assessed. This authority assessment may guide (narrow / expand) the information gathering process.

(c) On the basis of the collected available public information – and partly guided by the authority of the model owner – a decision is taken to obtain one or several models for detailed evaluation.

(d) The model is evaluated for validity (and possibly many other criteria such as cost, practicality, usefulness, applicability etc.). Should the model be found to be unsuitable, further model requisitions and/or model research may be necessary. (The validation procedure may also affect the validator's opinion of the model owner.)

(e) After the most suitable model has been chosen, it is implemented e.g. as the basis for an information systems design or academic research project.

These steps are summarized in Figure 9-1, with the solid arrows indicating the normal course of events and the dashed arrows incidental, less important or optional actions and feedback loops.

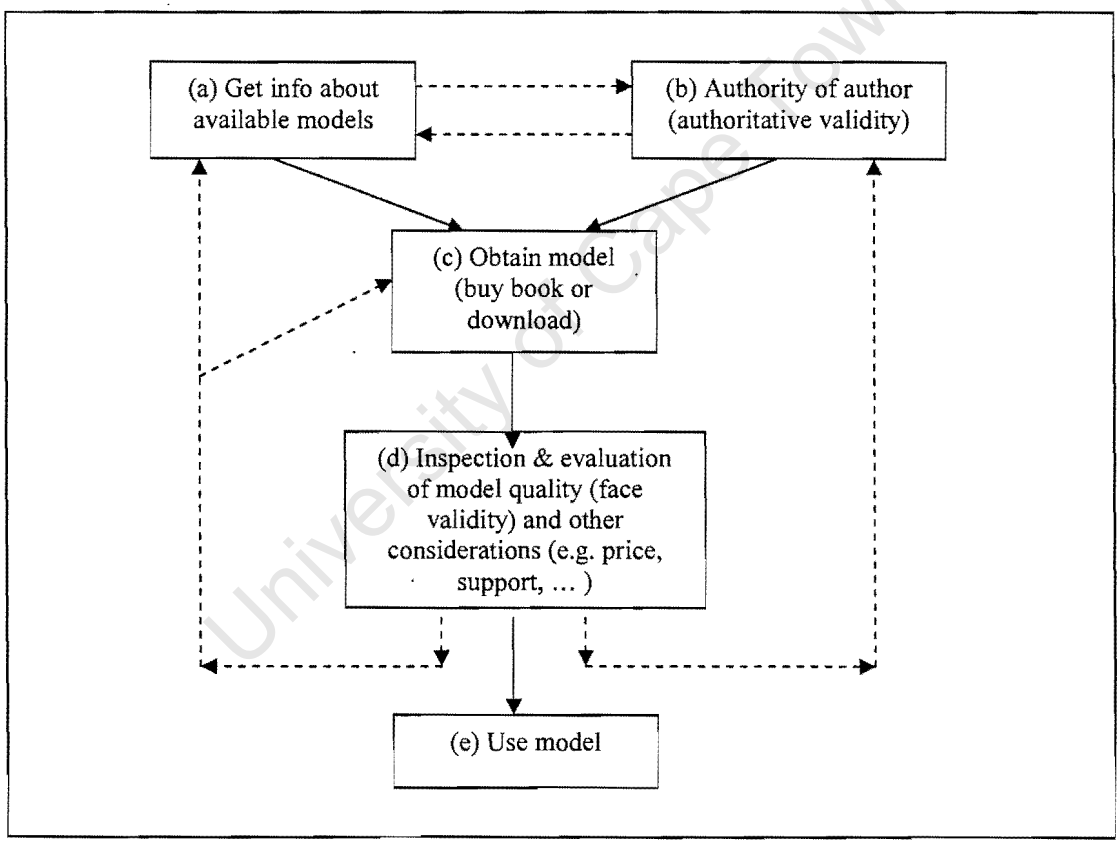


Figure 9-1: Illustrative Decision Model for Model Selection and Adoption.

The following proxies could be used to estimate some of the above factors:

- Cited publications by the lead author: implies both (b) authority and (e) model use (at least by academic researchers).
- Google ranking: implies (b) authority and (d) some type of evaluation by outside web pages.
- Amazon book sales ranking: implies (c) book (model) purchase.
- Informed estimate of (e) model use.

None of the proxies is ideal or definitive but they illustrate a practical way of estimating model validity. Further research may prove useful in refining the proposed "validation model".

9.1.1 Cited publications by the lead author

A search was done for the number of unique publications by the lead author (of the model or the institute/research group), as cited (referenced) by other publications in the field of computer and information technology. The database used is the online citations database provided by NEC Research Index aka Siteseer (http://citeseer.nj.nec.com/cs) and the search was conducted on 5 March 2002. The NEC Research Index is by no means the most exhaustive literature reference database, but it is the most appropriate to this research since it contains a full-text version of (virtually) all Adobe and PostScript research publications available on computer and information technology on the Internet. It uses the references in those publications to build up its database. It also includes advanced referencing analysis capabilities, arguably superior to that of any other electronic database. It is therefore considered to be far more authoritative and representative than many other databases.

It was not straightforward to determine which publications belong to the lead author (e.g. searches for "J Miller" or "T Williams" each turned up more than 1000 citations). Even after elimination of duplicate citations, the total number of references generated by the database for all model authors (where the search string was constrained to the author field by "surname and "initial" or "first name") added up to 1726 listed documents. A next elimination was the removal of duplicate references that were not recognized as such by the AI algorithm in the database. Finally, only those references where the title was somewhat related to the modelling effort and activities of the author were considered. This is particularly relevant for those authors whose research history spans several decades and whose research focus has changed substantially (e.g. that of Mark Fox).

In total, 254 model-related references were found for 10 model "authors". The following models had a lead author with zero references: USB, Silverston, Baan, Random, Semi-Random. No lead author could be identified for the following models: Nippon, BelgAcc, AKMA, NHS, SanFran and both Ottawa models; these models are not developed by individuals but have a "corporate titleholder". Table 9-1 summarizes the results:

Table 9-1: Authority of Model Author Measured by Academic Referencing.

Model	Lead author	References
CYC	Doug Lenat	59
TOVE	Mark S. Fox	47
SAP	August-Wilhem Scheer	30
Fowler	Martin Fowler	25
Inmon	Bill/William Inmon	24
AIAI	Mike Uschold	20
ARRI	Don Liles	19
Purdue	Theodore Williams	17
Miller	James Grier Miller	10
Hay	David C. Hay	3
BOMA	Chris Marshall	0
Baan	Jan Baan	0
USB	Jean-Paul Van Belle	0
Silverston	Len Silverston	0
Random	Jean-Paul Van Belle	0
SemiRandom	Jean-Paul Van Belle	0

The lead researcher of the CYC model, Doug Lenat, has the largest number of cited publications. This project happens to be the largest pure research project on the list (excluding the models underlying the ERP systems, namely SAP & Baan).

The numbers seem to be indeed representative of the academic standing of the lead authors. Doug Lenat, Mark Fox, A-W. Scheer are all recognized leading academics. Martin Fowler and Bill Inmon are practitioners with a number of publications on their record. Mike Uschold, Don Liles, Theodore Williams and James Miller are respected academics with a good publications record. David Hay, Chris Marshall, Len Silverston and Jean-Paul Van Belle are relatively unknown (for their publications) in academia.

However, the validity of the measurement has a number of problems. Chris Marshall is widely respected in the practitioner's community and he partakes in a number of important industry forums such as the OMG.

The publications records of the authors are also skewed by the relatively large number of co-authored papers, although this actually applies to virtually all authors. Since a co-authored paper adds almost equally to a person's standing and experience, it was decided not to weigh the references by a fractional weighting factor (i.e. one divided by the number of authors). Practitioners have a different publications profile (quality, type and number of publications) than academics. Publications profiles even differ between reference disciplines.

A more valid analysis, therefore, is to compare author references for models within the same reference disciplines.

- TOVE scores more than twice as high as AIAI, and TOVE is indeed widely regarded as the more formal and developed enterprise ontology. CYC is again a much larger research effort than either, but it must also be remembered that its scope is much wider than the enterprise domain. The number of researchers working on each of the ontologies is also in the same order as indicated by the values.
- Interestingly, SAP has, through Prof Scheer, indeed a much more academically sound research foundation than Baan, as reflected in their scores. This ranking also reflects the difference in the number of engineers who worked on the model.
- However, the lowly ranked Hay, BOMA and Silverston models are indeed not well validated (with Hay probably the best accepted in the market).
- Fowler's patterns have indeed been widely referenced and been validated in a number of forums.
- However, the model by the very productive data warehouse prophet Inmon is not as well-validated as the high number seems to indicate – quite the contrary in fact!
- The validity of the Purdue model is probably much higher than that of the ARRI model. The former model went through a number of standards-setting committee reviews whereas the latter appears more to be one of the many practice efforts by the ARRI researchers.

Overall, it appears that the metric indeed measures and ranks the academic standing or authority of the lead author associated with the model fairly accurately, especially within the same reference disciplines. There also seems to be a fair correlation with the amount of model validation that has happened in the academic community, except for the "practitioners systems

engineering" models in the books published by Hay, BOMA and Silverston where the validity would stem from their practical experience rather than from academics' feedback.

9.1.2 Books-based Models and Amazon.com ranking

Models must be obtained before they can be inspected and, possibly, a decision is taken to use the model. No systematic information is available on how many individuals inspected a given model, but a rough estimate can be made by the following two measures:

- For models available in book or document format, the number of copies distributed or purchased is a close indicator. A problem is that many documents that are distributed free of charge or purchased by corporate institutions may not necessarily be read by anyone. On the other hand, many books, including those on which an individual spent a significant amount, and books available for lending in e.g. libraries, may have multiple readers.
- For models available on the Internet, the number of downloads is an indicator similar to book distribution. Again, only the primary download can be measured. Subsequent private copying and distribution of the source files cannot be ascertained, nor can it be established whether the downloaded files are actually inspected. Search engines and other web-bots compound the problem with false hits.

Unfortunately, even these indicators cannot be reliably obtained. Book sales are often not made public by the respective book publishers/authors and web site traffic statistics are typically available only to the site owners, and many of these probably do not actually track downloads or page hits. None of the websites which were visited had (public) page counters, which could have been used as an alternative indicator.

A next best proxy is to use the sales figures by a large, representative book seller (such as Ingram's). Unfortunately these statistics are also not available for public inspection. The only readily accessible resource is Amazon.com's sales ranking feature that allows one to check the relative sales ranking for selected books. JungleScan (previously trading under the name of AmazonScan but changed its name due to trademark infringement) allows one to track a portfolio of several titles simultaneously.

Figure 9-2 a screen print of the results for the model source books available on Amazon.com. Note that Miller's book was not an Amazon available title.

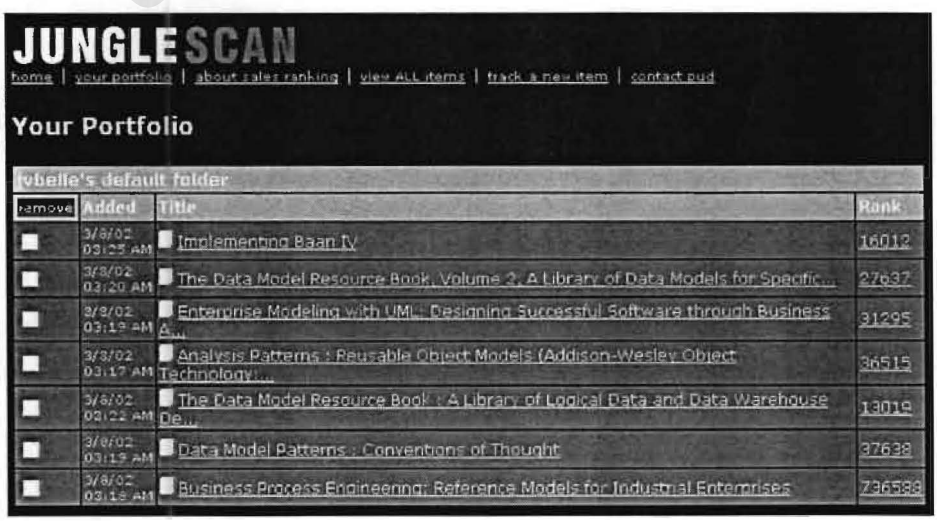


Figure 9-2: JungleScan Tracking of Amazon Sales Ranking for Portfolio of Model Books.

Table 9-2 summarizes the book sales ranking for the 7 models which were tracked.

Table 9-2: Authority of Model Books by Amazon Sales Ranking.

Model	Reference	Date of publication	Sales Rank on 8-Mar-2002	Sales Rank on 13-Jan-2003
Silverston (1st / 2nd ed.)	[SILV97; SILV01]	1997 / 2001	9852 / 13652	13019 / 27637
Baan	[PERR98]	1997	12175	16012
BOMA	[MARS00]	2000	95055	31295
Fowler	[FOWL97]	1997	27925	36515
Hay	[HAY96]	1996	34589	37638
SAP	[SCHE98]	1994 (1998)	819742	736588

Table 9-2 lists the Silverston model as the leading seller, especially if the fact is taken into account that two different editions are featuring first and third respectively. The Baan model comes a clear second. The BOMA model was in third position in the most recent check, although it has moved up significantly during the period under consideration. Fowler and Hay lie close together in fourth and fifth position, not far behind BOMA. SAP comes very definite last with a huge gap.

The validity of these figures, however, is subject to some serious methodological concerns:

- The date of publication may seriously distort the sales rank figure – this could explain the big move for BOMA.
- The figures for Baan and SAP are very misleading: The Baan publication [PERR98] is a guide to Baan implementation by an American publisher whereas Scheer's book is a much more academic publication on the reference model itself by a Germany-based publisher. In fact, the SAP books are much more popular than the Baan books as revealed by the following investigation. An Amazon.com search was done (on 13-Jan-2003) for the different book titles containing SAP or Baan. A total of 244 different SAP software-related titles was found (excluding another 46 titles with "sap" in the title but not related to software; "sap" is also a Thai word). In contrast, only 14 different book titles were on Baan (excluding another 14 mainly Dutch titles not related to Baan software, and none by Jan Baan who established the Baan software company). The availability of these different books clearly indicates a huge popularity lead of the SAP ERP software, though not necessarily in terms of its model.

Overall, the Amazon.com sales rank appears to be valid only in terms of the ranking of the Silverston, BOMA, Fowler and Hay models.

9.1.3 Online models and the Google PageRank™

As described above, ideally, web traffic analysis would be available to measure the popularity and download (inspection) of the large number online of enterprise models. Unfortunately, those statistics are often not collected, and never publicly available (unless the web page contains a page counter). In principle, it is possible to sample the web traffic in a manner similar to a "book seller sales ranking" by scanning the IP traffic going through one of the larger Internet routers. Apart from the practical, ethical and legal issues involved, the proximity of the Internet router to the web server undoubtedly also plays a large role and presents a methodological problem. Hence web traffic analysis remains a practical impossibility.

However, the existence of web links to an online model constitutes a possible proxy for both the authority and, to a much lesser extent, the positive evaluation (validity assessment) of that online

model. Although an accurate count of the world-wide number of links to each web site is impossible, the Google search engine provides a rough but interesting indicator of the number of (outside) links to a particular web page. The PageRank™ system is a proprietary system developed by the founders of the Google search engine which is used to measure the popularity (external rating) of a web page by counting the number of hyperlinks to the page [LEVY02]:

> "PageRank relies on the uniquely democratic nature of the web by using its vast link structure as an indicator of an individual page's value. In essence, Google interprets a link from page A to page B as a vote, by page A, for page B. But, Google looks at more than the sheer volume of votes, or links a page received; it also analyses the page that casts the vote. Votes cast by pages that are themselves "important" weigh more heavily and help to make other pages "important."
> [http://www.google.com/technology/]

Table 9-3 lists the Google PageRank™ for both the organizational URL (a proxy for the *authority*) and the actual URL (as given in Chapter 6) from which the model can be downloaded or accessed (a weak proxy for the *validity* of the model) as of 8-March-2002.

Table 9-3: Authority of Web Model and Author Using Google Rank.

Model ID	Organization URL	Google PageRank™ for model URL	Google PageRank™ for organization URL
SanFran	www.ibm.com	5	9
SAP	www.sap.com and www.sap-ag.de	NA	8
AIAI	www.aiai.ed.ac.uk	7	7
CYC	www.cyc.com	6	7
Fowler	www.martinfowler.com	NA	7
Baan	www.baan.com	NA	7
Ottawa-Big	www.ottawabusinessjournal.com	6	6
Ottawa-Dense	www.ottawabusinessjournal.com	6	6
TOVE	www.eil.utoronto.ca	4	6
Purdue	www.pera.net	4	6
Inmon	www.billinmon.com	4	6
NHS	www.nhsia.nhs.uk/def/home.asp	4	6
Nippon	www.nsc.co.jp	NA	6
Hay	www.essentialstrategies.com	5	5
AKMA	www.akma.com	5	5
ARRI	arri.uta.edu (/enteng)	3	5
Silverston	www.universaldatamodels.com	0	5
USB	www.usb.sun.ac.za	NA	5
BOMA	www.sesh.com	NA	4
Random	www.commerce.uct.ac.za/~jvbelle	NA	4
Semi-random	www.commerce.uct.ac.za/~jvbelle	NA	4
BelgAcc	NA	NA	NA
Miller	NA	NA	NA

The following are some tentative observations from the above table:

- Models sponsored by large multinationals such as IBM (SanFran) and SAP exude significant "authority", although this is not necessarily reflected in, and is quite distinct from, the evaluation of the individual model. For example, compare SanFran's ranking of 5 against its owner's ranking of 9; or ARRI's corporate rating of 5 as opposed to the low rating of 3 for its model.

- Without fail, the model evaluative ratings are lower than or equal to those of the organization, hence they should probably not be compared.
- Academic models tend to have an unexpectedly high ranking viz. the AIAI and CYC models. This may be influenced by the academic nature of many sites and the relative prominence of computer science (and the sub-discipline of AI) on the internet.
- The PageRank™ appears to validate two relative model rankings within the same disciplines:
  - SAP is rated higher than Baan (organization)
  - AIAI is rated above TOVE (despite its bigger size and claims to better formality).

There appears to be little interpretive value for the remaining models.

Overall, the validity of the measure appears fair to good. Google's PageRank™ provides an interesting perspective on the perceived authority of the sponsoring organization within the web community, so seems to measure the perceived authority of the model sponsor or author. It can be collected fairly easily, regardless of the medium in which the model is distributed. Unfortunately, the measure does not appear to measure the actual model validity very well.

With respect to the validity of the measure, it must also be noted that the rankings can change over time. Though the rankings for the model URLs remain fairly static, those for the organizational URLs change much more rapidly e.g. by 13-January-2003, the PageRank™ for SAP had increased to 9 and the one for CYC to 8.

9.2 Availability and cost

An important pragmatic aspect is the availability, including access, and cost of the models. Because of the sampling methodology used, i.e. only models that were essentially free were included in the testbed or sample of models, all models have a low cost and were fairly easily accessible to the general public. Table 9-4 gives a more specific overview of current model availability, physical size, cost and whether the model is available in digital format. The following discusses each of the criteria in more detail.

9.2.1 Model availability

Model Availability discusses the medium and state of availability i.e. whether it is a download from a public website or a published book, and if it is still available. It is interesting to note that two websites have removed their models since the date on which they were initially selected for the model database: IBM's San Francisco framework as well as Ottawa's Business Journal's business lexicon (which formed the basis of the Ottawa-Big and Ottawa-Dense models). A new version of the initial (Open)CYC model, which was not available during the last three years of the research project, has now been publicly released as from 17th December 2002. Miller's book on living systems is no longer in print but can be found in many libraries.

It can be concluded that model availability is not a real issue or differentiating factor for the models used in this study, apart from those models indicated. All models, including the few models which have been sourced from less accessible research reports, and those that have disappeared from the web, will be made available as part of this thesis and from my website in XML format.

Table 9-4: Model Availability, Size and Cost.

Model ID	Availability (as of 31-Dec-2002)	Size (Physical)	Costs	Capture time (days)	Available in digital format?
AIAI	Public download	small <100K; 85KB in KIF format	Free	1 ½	Yes
AKMA	Public download	1.3 MB text & GIFs	Free	2 ½	No
ARRI	Public download	132KB (pdf format)	Free	1 ½	No
Baan	Book in print	576 pages	Book: $59.99 ERP expensive	3	Yes
BelgAcc	Legal publication	32 pages	Free: legal standard	1 ½	No
BOMA	Book in print	1.7MB repository dump; 30.2MB docs & source	Book: $44.95	2 ½	Yes
CYC	OpenCYC toplevel 6000 concepts v. 0.7.0 rel.17-Dec-02	42MB database (electronic version excluding tools)	OpenCyc: Free; ResearchCyc: Free; Full: $10mln	1 ½	Yes
Fowler	Book in print	356 pages	Book: $49.99	2	No
Hay	Book in print	268 pages	Book: $55.95	3 ½	No
Inmon	Public download	480KB (gif format)	Free	2	No
Miller	Book (out of print)	1082 or 154 pages	Free / Book: $68:00	2	No
NHS	Public download	23.1MB incl images	Free	2	Yes
Nippon	No access	1 page	Free	½	No
OTT.-Big	Was public; now off web	348KB text file	Free	½	Yes
OTT.-Dense	Was public; now off web	129KB text file	Free	½	Yes
Purdue	Public download	10.7 MB (pdf)	Free	2	No
Random	Research document	118KB text file	Free	½	No
SanFran	Was public; now off web	2.86MB (full file set)	Free	2 ½	Yes
SAP	Book (out of print)	770 pages	Book: $48.00; ERP Expensive	3 ½	Yes
Semi-Random	Research document	129KB text file	Free	½	Yes
Silverston	Book in print	355 pages	Book: $55.00 CD-ROM: $300.00	3	Yes
TOVE	Public download	1 MB html files; 7.83 MB all files	Free	2	Yes
USB	Research document	618Kb .xls format	Free	2	Yes

9.2.2 Physical Model Size

The physical size of a model refers to the amount of space taken in its native format in the original medium, where possible in electronic medium. This is very distinct from the model's *logical size* as measured in number of entities and relationships, and as was discussed under syntactic analysis. Although storage should not be a problem for any of the above models, this is still an important statistic since it may theoretically influence its download times e.g. downloading the full OpenCYC model via a 28.8K modem would take about three-and-a-half hours. For books, it may reflect costs of shipping, scanning or faxing. The two options for Miller's book reflect the total number of pages in his entire book or just the Chapter (10) discussing the organization.

Note that the model sizes tend to be relatively small for all models – no model is larger than 50MB. Model size in bytes is very dependent on the format: a number of models is available in GIF format which tends to inflate their apparent size. A model in KIF or TXT format is, relatively speaking, much more compact. Model size alone is not the main determining factor for copy or transfer speed. For instance, although the BOMA model is only 30.2 megabytes, it consists of 4205 files in 219 separate folders. Making a backup of the BOMA model from one hard disk to another took about 10 minutes whereas copying 4 files of 8 megabytes each (totalling 32 megabytes) took less than 5 seconds.

Overall, it can be concluded that physical model size is not an important pragmatic issue or differentiating factor in a corporate or academic context.

9.2.3 Model Cost

Where model cost becomes a real issue, it is theoretically possible to calculate such ratios as *cost per model element* (or e.g. *cost per entity)*, although it is strongly suspected that this will not really be meaningful in the light of all the other decision variables such as overall quality, support etc. It is clear that a pre-packaged model can save a lot of valuable time and arguably give extremely high return on investment ratios. On the other hand the study and customization involved with adopting a packaged model can be almost equally expensive. This can be compared to the debate surrounding packaged off-the-shelf software versus in-house development.

Because extremely limited research funds were available for this research, none of the expensive commercial models was tested. Therefore cost is not a pragmatic issue for any of the models in the database.

In practice, enterprise models range from cheap to very expensive. The following are some examples:

- The "Visible Universal Model" developed by Visible Systems Corporation, and marketed as a "Universal Enterprise Information Architecture", costs $100,000.
- The cost for the full CYC model is unavailable, but the 10 consortium partners contributed US$1 million each for a period of 10 years, in return for privileged access to the knowledge base.
- The Public Petroleum Data Model is available to members only. 2003 membership fees range from US$445 to US$23,100 per annum [http://www.ppdm.org/ftp/public/misc/2002-03_Fee_Schedule.pdf].
- Both the SAP R/3 and Baan models are normally released only in electronic format (with the accompanying tools) to those organizations who buy their ERP systems. A medium-sized implementation of SAP R/3 ERP software will cost between £2000 and £3000 per user to buy [ANON99].
- ADRM specializes in prepackaged enterprise models. Although it does not reveal any pricing details on its website, and an enquiry went unanswered, it is in the region of several thousands of dollars.
- Wizdom.com's generic Manufacturing Enterprise Reference Model costs US$ 495 (CD-Rom version).

9.2.4 Availability in Digital Format.

One of the drawbacks of many of the models used in this research is the fact that they were not available in native digital format. Even where some models were available off the internet, a number of them was in scanned or graphics format, which was not amenable to digital processing i.e. Purdue, AKMA, Inmon and ARRI had to be printed out and re-entered. Refer also to the methodology comments for the SanFran and BOMA models which were available in Java source code, but the underlying logical model was easier to extract from the printed documentation. Finally, all the models in the books had to be captured by hand.

The capturing process represents a real cost in terms of person-days spent as well as being a potential source of inaccuracies. Hence the availability of the model in a computer-readable format is a definite pragmatic advantage for a model. For example, the Silverston model is available in book form at a cost of US$55.00, whereas the same model can be obtained in electronic format on a CD-Rom for US$300.00.

Although no accurate record was kept for individual models, the data capturing alone of *all* the models for this research took 43 person-days or about 340 hours. This figure excludes the considerable extra time required to locate, source and prepare the models for capture, since this was done over a period of several years. However, the figure does include the conversion of the electronic models to the standard meta-model format used for this research. Depending on the value placed on the researcher's time, the total capturing and conversion cost for all models can thus be estimated at between R 20 000 and R 40 000, i.e. an average of R1000 to R2000 per model. Table 9-4 gives approximate time investment (to the nearest ½ day or 4 hours) for each model.

9.2.5 Conclusion: Overall Availability

The models can be broadly grouped in a number of categories by combining the above factors. The somewhat subjective rating given is a composite of availability, accessibility and cost:

- Excellent: Silverston, BOMA.
- Very good, some conversion (and selection time) required: CYC
- Good, but capturing required: AKMA, ARRI, BelgAcc, Fowler, Hay, Inmon, Nippon, Purdue, SAP.
- Good, but non-trivial conversion required: AIAI, NHS, TOVE, USB.
- Some difficulties: Baan, Miller.
- No longer available from the source: SanFran, Ottawa.

9.3 *Flexibility and Adaptability*

9.3.1 Description

Flexibility refers to the extent to which models can be changed or adapted to different situations. This flexibility depends on a number of factors. The following criteria are believed to be the main determinants of model flexibility, based on the following simplistic process of adapting models.

1.Capture Model → 2.Change ("Adapt") Model → 3.Implement ("Adopt") Model

The first factor checks how easy is it to load the model for electronic (tool-based) manipulation. This is measured here by checking whether the model is available in digital format. This attribute was also considered above as a determinant of model availability.

The second factor is whether the model is customisable, extendable and/or reusable. There is a difference in meaning between the three terms:

- **Customisability** refers to the ability to change existing constructs to suit differing circumstances.
- **Extendability** refers to how easy it is to add *new* (usually more specialized) constructs and implies, for example, that the hierarchical structure is clear and well-designed.
- **Reusability** refers to how easy it is to use the same model for different purposes and requires, *inter alia*, proper documentation (see below) as well as genericity of application.

In the context of software packages, computer programming code or hardware components, there is a clear distinction, and large degree of independence, between these three constructs. However, for models, these three concepts generally overlap to a large degree. It is important to realize that generic enterprise models are almost by definition customisable since the overall purpose of these models is to apply them to specific situations. Any of the models in the database can, in principle, be customized or changed. So, in this context, the extent to which this customisability has been engineered into the model as a conscious decision is more relevant.

Finally, as the third flexibility measure, the implementation independence refers to how dependent the model (in its source format) is on the availability of a particular platform, in particular whether proprietary application software is required.

Note that there are also other factors which influence a model's flexibility. For instance, the amount and quality of the documentation affects reusability and change. High-quality model documentation will include some of the reasoning behind why a particular situation was modelled in that way, as well as suggestions for implementing the model. For example, Fowler, SanFran and BOMA have been specifically designed to be customized, and pattern description languages include specific slots for the rationale behind a pattern, anti-patterns, related patterns and implementation issues. Other important flexibility considerations are the use of inheritance concepts (super-classes, multiple inheritance), the degree of formality and the overall model complexity. Since these are discussed in significant detail elsewhere in this research, they will not be considered here again.

It is important to note that flexibility as defined above is very distinct from the concept of genericity. For instance, Miller's model can apply to any living system, including a biological cell or a national state! It is thus extremely generic, and a popular interpretation of the word "flexible" could therefore apply. However, in this research, the term flexibility has been reserved to the extent in which the model *in its original specification and format* can be applied to specific real-world organizations. For example, Miller's model is not available electronically, it is specified very informally, it has no supporting tool etc. It is therefore not really flexible from a pragmatic or practical point of view.

9.3.2 Measurement

An attempt was made to calculate a quantitative measure for the overall flexibility of a model, which would allow the ranking or classifications of the models. It was therefore necessary to allow a limited number of categories for each of the factors, and associate these with a value.

- Digital availability takes a binary value Yes / No, with respective associated numeric values of 1 and 0 for purposes of calculating the *flexibility score*. Note that a value of "No" is awarded to a model such as Inmon's which has the diagrams in electronic format (in .GIF files) but the underlying model elements (the entities and their relationships) cannot be extracted electronically from the graphics files.

- Customizability, extendibility and reusability were also rated along a Yes / No scale, but with some intermediate options. A strong underlying criterion in awarding a value was whether the model was purposely and inherently designed to be extendable. For example, the BelgAcc system has allocated specific account codes for user-specific needs. The customizability of a model also depends on the public availability of modelling tools by which the model can be manipulated in its native format. Note that all models can be imported into any reasonable modelling tools using the XML model database. The specific categories and their associated numeric values are the following.

No	Limited	Yes / some	Yes
0	1/3	2/3	1

- Finally, the degree of independence of implementation platform was rated from low to high and awarded values as follows.

Low	Medium	High
0	½	1

9.3.3 Analysis

Table 9-5 lists the considered ratings for each model against the three flexibility factors. Since the choice of tool as well as the modelling notation influences the customisability, this information has been included as a separate column.

Featured at the top are, not surprisingly, the models from the data modelling and patterns background, although Fowler and Hay lose out because they are not available in electronic format. The ERP-based models also rank high despite their need for specialized tools. Somewhat surprisingly low scores are obtained by Inmon, AKMA, and ARRI. A low rating is a special concern for the Purdue model, since it is purported to be a CIM standard. This should be contrasted to NHS which, also a proposed standard, is much easier to adapt (and therefore to adopt!).

9.4 Model Currency and Maturity

A critical quality of any model is how well it reflects the underlying domain. If the domain changes, the model needs to be updated to reflect these changes. Thus an important pragmatic aspect of a model is how often it is updated, what the update policy (including version control) is and when the last update or change occurred.

Table 9-5: Flexibility, Adaptability Customizability and Implementation Independence.

Model ID	Available in digital format?	Tool used	Customizable / extendable / reusable	Implementation independence	Overall flexibility score
SanFran	Yes	UML & Java	Yes	High	3
Silverston	Yes	EERD modelling tool	Yes	High	3
BOMA	Yes	UML & Java	Yes / some	High	2 1/4
Baan	Yes	DEM	Yes / some	Medium	2 1/4
CYC	Yes	Proprietary	Yes / some	Medium	2 1/4
Ott.-Big	Yes	(text)	Limited	High	2 1/4
Ott.-Dense	Yes	(text)	Limited	High	2 1/4
SAP	Yes	ARIS	Yes / some	Medium	2 1/4
BelgAcc	No	Accounting package	Yes	High	2
Fowler	No	OO modelling tool	Yes	High	2
Hay	No	EERD modelling tool	Yes	High	2
TOVE	Yes	Ontolingua/KIF	Some	Medium	2
Semi-Random	Yes	(text)	Limited	Medium	1 3/4
AIAI	Yes	Ontolingua/KIF	Some	Low	1 1/2
NHS	Yes	System Archictect	No	Medium	1 1/2
Random	No	(text)	Yes	Medium	1 1/2
USB	Yes	Lotus 1-2-3	Limited	Low	1 1/4
AKMA	No	System Architect	No	High	1
Inmon	No	(drawing tool)	Some	Medium	1
Purdue	No	(drawing tool)	Limited	Medium	3/4
ARRI	No	IDEF0 modelling tool	No	Medium	1/2
Nippon	No	(drawing tool)	No	Medium	1/2
Miller	No	NA	Limited	Low	1/4

Unlike for most domains, however, the "generic enterprise" domain does not change dramatically over time: organizations today are structurally very similar to those of a couple of decades ago. Thus the currency of an enterprise model is less of a quality issue in this research than for most other models. (Note that an enterprise model for a *specific* (individual) organization is much more volatile.) For example, Miller's Living System Model, the oldest model under consideration, is still as applicable now as it was in 1978. This does not mean that the "generic enterprise" domain remains fully static e.g. the enterprise models for the ERP systems (Baan, SAP) are continuously updated to accommodate new business models (e-commerce) and become more generic (service industries, government); the BelgAcc model has been revised several times to include additional information and new accounting procedures.

Closely related to the versioning and updating of a model is the maturity of the model, with specific reference to its stage in the model development life cycle. Because of the iterative nature of the modelling process, rough "first-cut" models may be created, to be refined at later stages. Generally, the more revisions, the less likely future changes are to the model. In this research, the likelihood of substantive changes to the model when next revised is referred to as the maturity of the model: the less change, the more mature the model. In common parlance, maturity also incorporates a qualitative aspect in terms of the model validity, but this was discussed above and is therefore not considered here.

Table 9-6 details some facts related to currency and maturity for each of the enterprise models. Based on an attempt to categorize the models in the above table, the following groups of models, along with recommended "somewhat whimsical" labels, are suggested. Note that this stereotyping by necessity resulted in the creation of a somewhat caricature effect.

Table 9-6: Model Currency and Maturity.

Model ID	Maturity	Date last changed	Updates & version control
AIAI	Final, revised	1998	No more
AKMA	1st version	2000	None so far
ARRI	Final (1st version)	1990	No more (last web update: 1998)
Baan	Mature, many revisions	2002	Regularly
BelgAcc	Very mature	1998	Regularly extended but stable
BOMA	1st version	2000	None so far
CYC	Long track record, institutional support, untested in real world apps	2002	Yes, including planned versions
Fowler	1st version	1997	None so far
Hay	1st version	1996	None so far
Inmon	Final (1st version)	2000	Has not been updated
Miller	Final (1st version)	1978	No more
NHS	Final (1st version)	1999	No more
Nippon	Final (1st version)	1987	No more
Ottawa-Big & Ottawa-Dense	Final? (1st version)	2001	Disappeared from website - last seen on 12-Nov-2002
Purdue	Final (some revisions)	1991	No more
Random	Final	2002	No more
SanFran	Final (version 2.1)	2000	IBM no longer supporting it: memo 201-212 , 31-July-01
SAP	Mature, many revisions	2002	Regularly
Semi-Random	Final	2002	No more
Silverston	2nd version available	2001	2nd edition - minor revisions and some additions
TOVE	Under development, limited QC	1999	New sub-ontologies may be developed (quality was the last)
USB	Final (1st version)	1988	No more

- "**Toys**": Models created for the purpose of playing, to be discarded as soon as its usefulness has been outgrown. They are characterized by thier ad-hoc nature and have limited use for anyone else. This refers to the two random and two Ottawa models which were developed specifically as validity or testing benchmarks for this research. (Note that the "toy" label does not apply to the underlying source dictionaries from which the models were derived!)

- "**Fossils**": These are the oldest models, more than a decade old, which have typically never or very rarely been revised: Miller, ARRI, Purdue and Nippon. They are typified by a relative obscurity and the use of fairly dated modelling techniques. They are not available in electronic format.

- "**Souvenirs**": Slightly more recent models, typically conceived as an academic exercise and mostly forgotten due to lack of research funds or interested research staff, although still fondly remembered by some academics and featuring strongly on their (or their institution's) publication and project record. They are available in electronic format, along with some relatively sophisticated tools for their manipulation – often required due to the use of an obscure (outside the academic community, at least) modelling language. This is an intermediate stage, often as a prelude to becoming a "fossil". Prototypical models in this category are AIAI and USB. Border-line cases are TOVE and CYC although there is still some activity happening for both of these.

- "**Trophies**": Models usually published in book form, summarizing and show-casing the experience of a consulting company or modelling expert, often with an explicit or implicit

invitation to contact the author for consulting work or at least in an attempt to establish the author's modelling credentials. Most of these models are unlikely to be revised in the future but use a fairly common and recognized modelling language such as EERD or UML. The models in this category are AKMA, BOMA, Fowler, Hay, Inmon, SanFran and Silverston; NHS can be considered a borderline case.

- "**Stars**": The exclusive group of living, working models that are widely used in the community and need to be updated fairly often. The obvious examples are the ERP models SAP and Baan, although BelgAcc also qualifies for entry in this group. Some fossils and souvenirs may have had (Purdue) or still have aspirations (TOVE? CYC) to be admitted.

9.5 Support

Before considering the adoption of a model, the issue of support is often an extremely important consideration.

Support includes the following dimensions:

- **Tool** support. What tools exist to manipulate the model in its original format? Are these tools platform specific? Proprietary or publicly available?
- **Vendor** support. Does the author or supplier of the tool offer assistance? Although some of this may be offset by sufficient documentation, the ability to get online or telephonic support may be crucial in certain contexts.
- **User** base. If there is a large and/or enthusiastic installed user base, this creates a community where problems, examples and other information can be shared. Where there is a large user base, an online forum is often available for users to share experiences and ask questions.

Table 9-7 lists some facts for the various support types. Using the "somewhat whimsical" model classification introduced in the previous section, one could summarize the above table as follows:

- Support for *fossils* is as good as non-existent. You are on your own.
- *Souvenirs* (and presumably the *toys*) have very limited support insofar as the original researcher or research group is still available and could possibly be enticed with a research grant. You are likely to have to master a fairly obscure, beta-version quality modelling tool.
- Support for *trophies* is available at consultancy rates but you can use standard modelling tools which are commercially available.
- *Stars* have wide user communities, and you can get lots of help from third-party vendors, including textbooks and training courses. Be prepared to use far more expensive, professional, but unique tool sets

Table 9-7: Model Support by Tools, Vendor and User Base.

Model ID	Tool support	Vendor support	User Base
AIAI	Ontolingua/KIF	None	Unknown, assumed to be very limited
AKMA	System Architect	High (consultant)	Unknown
ARRI	IDEF0 modelling tool	No	Unknown, assumed to be very limited
Baan	DEM	Excellent	2 800 customers in 1999 [MEND00, p. 7]
BelgAcc	Accounting package	Legal	Extremely wide: all companies in Belgium.
BOMA	UML & Java	Yes - consulting SESH	Unknown
CYC	Proprietary and some open source	Yes: 2-tier support level with tech support, consulting services, training etc.	Some. 6 mailing lists & bug/project tracking; open source community; official SourceForge project
Fowler	OO modelling tool	No	Unknown
Hay	EERD modelling tool	No	Unknown
Inmon	(drawing tool)	Yes: consulting	Unknown, derived from practice but probably fairly limited
Miller	NA	None	Two papers
NHS	System Archictect	NHS Information Authority /Healthcare Modelling Program	Unknown
Nippon	(drawing tool)	None	Unknown but assumed to be very limited
Ottawa-Big/Dense	(text)	None	None
Purdue	(drawing tool)	Some: institutional / standard	Unknown
Random/ Semi-Rand	(text)	None	None
SanFran	UML & Java	No longer - superceded by WebSphere	Unknown
SAP	ARIS	Excellent	11 000 customers = 20 000 installations in 1999 [MEND00, p. 7]
Silverston	EERD modelling tool	No	Unknown
TOVE	Ontolingua/KIF	None (except contract research)	Unknown though some made its way into the PIF standard
USB	Lotus 1-2-3	None	Unknown but assumed to be very limited

9.6 Business Context, Alignment and Goal

Most of the remaining considerations are likely to concern the *alignment* of the model with the context in which it is to be used, see e.g. [BRIN96, p. 226]. It is therefore important to consider the origins of the model as well as the original purpose for which it was developed. Depending on these, if models are to be used, say for the development of systems to be implemented in specific organizations, there may be significant problems with the transfer or integration with the exiting organization.

Table 9-8 lists some origin data and design considerations, with a subjective interpretation on how easily the model would fit into an existing business framework or architecture.

Table 9-8: Model Alignment, Background, Goal and Business Integration Impact.

Model ID	Theoretical foundation?	Effect on business / ease of integration	Reference discipline	Purpose / Goal
AIAI	Some	Difficult	Ontology	Academic
AKMA	No	Difficult if different industry	Data modelling / CASE	Data modelling
ARRI	No	Difficult	Enterprise Engineering	Academic
Baan	Yes / some	Significant impact!	Enterprise Resource Planning	ERP
BelgAcc	Yes (accountancy)	Easy	Accountancy	Accounts
BOMA	Some	Easy	Business Objects	Data modelling
CYC	Yes	Difficult	Artificial Intelligence	AI reasoning
Fowler	Some	Fairly easy	Object-oriented Patterns	Data modelling
Hay	No	Fairly easy	Data modelling	Data modelling
Inmon	No	Fairly easy	Data Warehousing	Data Warehousing
Miller	Yes: philosophy, & systems theory	Difficult	Systems Theory	Philosophical analysis
NHS	No	Difficult if different industry	System Engineering	Data modelling
Nippon	No	Difficult	Enterprise Engineering	Unknown
Ott.-Big	No	Little	Linguistics	Dictionary
Ott.-Dense	No	Little	Linguistics	Dictionary
Purdue	Yes	Some	Enterprise Engineering	CIM
Random	No	Little	Academic	Dictionary
SanFran	No	Fairly significant	Business Objects	
SAP	Yes	Significant impact!	Enterprise Resource Planning	ERP
Semi-Random	No	Little	Academic	Dictionary
Silverston	No	Fairly easy	Data modelling	Data modelling
TOVE	Yes	Difficult	Artificial Intelligence	Academic
USB	Some	Easy	Finance, System Dynamics	Financial modelling

Note that the above criteria cannot be interpreted as a ranking or even in terms of desirability. If the purpose of a model is to develop a conceptual understanding of organization, to do abstract academic research, or create some reasoning capability within a database, then the evaluation criteria will be fundamentally different from when the model is required to develop an actual information system.

9.7 Summary and Validation of Pragmatic Analysis

Pragmatic model analysis, as defined in the framework used for this research, was concerned with those metrics and criteria which cannot be assessed purely on the basis of the information contained within the model, but which requires the consideration of information regarding the use, environment or context of the model i.e. information outside the model. Most analysis relies on the searching and ranking of certain specific information details, often involving a degree of subjective interpretation.

Model ***validity*** is a complex and composite measure. Validity, "how well the model matches reality", is best represented by the concept of *content validity* and, in the absence of statistical and experimental measures, is best measured by means of *face validity* i.e. the acceptance of the model by practitioners in the field. Another pragmatic criterion used in the commercial world is referred to as *authoritative validity.* This can be measured by the number of author citations which measures and ranks the academic standing or authority of the lead author associated with the model relatively accurately, especially within the same reference disciplines. Depending on the publishing medium of the model, metrics that are reasonably easy to collect are the relative sales ranking for book-based models (e.g. by Amazon.com), and popularity of a web page by counting the number of external hyperlinks to the page for web-based models, e.g. using the Google PageRank™ system. Both measures exhibit some validity, but not without concerns.

Another important pragmatic aspect is the ***availability*** and includes accessibility, ***cost*** of the models, current model availability, physical size, and whether the model is available in digital format. ***Flexibility*** refers to the extent to which models can be changed or ***adapted*** to different situations. A composite flexibility measure can be used that incorporates the availability in digital format, tool support, customisability and reusability and implementation independence. Model ***currency*** and ***maturity*** refer to how often the model is updated, what the update policy (including version control) is and when the last update or change occurred. Based on an attempt to categorize the models using the above attributes resulted in the following, somewhat "whimsically" labeled groups of models: toys, fossils, souvenirs, trophies and stars. The rating of model ***support*** includes the following dimensions: tool support, vendor support and user base. Support was found to be closely correlated to the model currency and maturity typology.

There are many more pragmatic considerations. Some of these remaining considerations concern the ***alignment*** of the model with the context in which it is to be used. This requires an investigation of the theoretical foundation of the model, its effect on business or ***ease of integration***, its reference discipline and the model's purpose or ***goal***. These can be described and rated subjectively.

Table 7-13 below is an attempt to rephrases these conclusions by means of explicit reference to the validity criteria as specified in 4.2.3. Because of the less formal nature of most measures, it is more difficult to ascribe specific validity concerns that for the semantic or syntactic analysis.

Table 9-9: Summary of Pragmatic Criteria Validity Concerns.

Framework Criterion (& Section reference)	Section	Metric / measurement	Recommended?	Specific validity issues (refer 4.2.3) in respect of the model sample.
Validity: authority & user acceptance	9.1	Academic author citations	Y*	Fair content validity but only when used within a given discipline
		Book sales by JungleScan	N	Low criterion and internal validity
		Google PageRank for organisation/author	N	Low criterion, construct and content validity
		Google PageRank for URL	N	Low criterion and content validity
Flexibility; expand-ibility; adaptability	9.3	Composite flexibility score	Y	
Currency; maturity	9.3	Descriptive table	Y	
		Textual categorization	Y	Need for external validation, no construct validity
Purpose; goal; relevance; appropriateness	9.6	Descriptive table	Y	Subjective (low internal validity)
Availability	9.2	Medium & status	Y	But low discriminatory value within the sample base (low criterion validity)
		Physical model size	N	as above
Cost	9.2	Purchase and download cost	N	as above
Support	9.5	Tool & vendor support, user base	Y	Subjective (low internal validity)

The pragmatic model analysis concludes the empirical validation of the framework. It is now possible to review some analysis techniques that were found to be not applicable or feasible, and to engage also in some higher-level analysis of the overall framework.

Chapter 10: Further Framework Analysis

Having attempted to apply and validate the framework to a variety of enterprise models at the most detailed level possible, it is now appropriate to review the framework at a higher level. This chapter describes some additional analysis approaches which could be used within the framework, proposes an approach to apply some of the semantic analysis techniques to build a "consensual" or common-denominator enterprise model, and checks the validity of the framework in other areas of IS research.

10.1 Other Model Analysis Techniques

This section includes some model analysis techniques which could, in principle, have been performed or tested but were not performed due to methodological or validity reasons. They should be seen as examples of how alternative techniques fit quite naturally within the framework.

10.1.1 Architectural Temperature (Syntactic)

In a somewhat non-conventional article, Salingaros proposes a number of metrics to measure the architectural life and complexity of a building [SALI97]. His measures are derived from a thermodynamic context, drawing heavily on the concepts of temperature and entropy as well as building on Christopher Alexander's ideas (who also pioneered the concept of patterns).

It is suggested that, because models are also built structures, just like architectural artefacts, it is possible to use or define equivalent expressions for Salingaros' measures for models.

The overall goal of Salingaros is to measure the emotional impact of a building and to this purpose two intrinsic quality measures are introduced, both measured on a scale from 0 to 10:

- The architectural *temperature* T is a composite index capturing the degree of detail, curvature and colour of a building; and
- The architectural *harmony* H is also a composite measure looking at the internal symmetry and the degree of coherence between the various building elements. It is the opposite of the architectural *entropy* which equals (10 – H)

It is suggested that these two quantitative measures can be combined in different ways to give an idea of the building's subjective emotional impression as follows:

- The *life* of a building would be indicated by the product of the temperature and the harmony.
- The perceived complexity of a building would be the product of temperature and entropy.

Each of his composite indices are made up of five components, each with possible values ranging from 0 (very little), 1 (some) to a maximum 2 (considerable).

Table 10-1 briefly describes the components making up the temperature and harmony indices. The table also includes a suggested equivalent for measuring or operationalizing these measures for models. Note that two ways are suggested: one for models using the diagrams, and one more general approach which is not based on a pictorial representation of the model.

As can be seen from Table 10-1, mapping the architectural concepts to diagram equivalents appears to be a pretty straightforward exercise, especially for the harmony elements for which

there are many candidate "aesthetics" measures available as discussed in Appendix C. However, to find roughly equivalent expressions on a conceptual level for textual models is not very straightforward and requires some leap of the imagination.

Table 10-1: Architectural Elements of Temperature.

	Building architecture description	*Possible measure for model diagrams*	*Possible measure for non-diagram based models*
T_1	Intensity and smallness of perceivable detail	Amount of detail in the diagram	Average number of entity attributes
T_2	Density of perceived textual differentiations	The diversity of size and shape of diagram elements	Variability (variance or standard deviation) in the number of entity attributes
T_3	Curvature of lines and shapes	The proportion of non-straight line-segments used in the diagram.	Number of different relationship *types* used.
T_4	Intensity of colour hue	The intensity of colours used (if any) in the diagram	The average number of different attribute *types*
T_5	Contrast among colour hues	The contrast between diagram colours or, if black & white, the amount of black/grey/reverse print	The variability or distribution of different attribute types.

Table 10-2: Architectural Elements of Harmony.

	Building architecture description	*Possible measure for model diagrams (as defined in Appendix C*	*Possible measure for non-diagram based models*
H_1	Vertical reflectional symmetries on all scales	BM; $SYM_{vertical}$ and/or $SYM_{horizontal}$	Degree of balance or symmetry in the hierarchical inheritance structure or fan-out distribution
H_2	Translational and rotational symmetries on all scales	SYM_{radial}	Skew-ness of the fan-out distribution
H_3	Degree to which distinct forms have similar shapes	PM; ECM; n_{shape} and/or UM_{form}	Occurrence of patterns; degree of chunking (grouping).
H_4	Degree to which forms are connected piecewise	RM; SMM	Average fan-out
H_5	Degree to which colours harmonize	BM (calculated using c_{ij}, only) and/or n_{colour}	Variability (standard deviation) of fan-out

The analysis has not been included in the framework for the following reasons:

- The validation of the original measures within an architectural context is still unclear. Although Salingaros has published a number of refereed papers on this and related topics, the lack of external references to his work cast some doubt on whether his ideas have been accepted in mainstream architecture.
- The application of the measures to the analysis of models appears possible, but it is not without methodological problems, especially for the purely syntactic analysis of models that are not diagram based.
- Even if the translation of the individual components could be supported from a methodological point of view, it is not clear what quality or aspect the overall composite indices would measure. What does the temperature or liveliness of a model represent? Should there be an emotional impression of a model? The place and intrinsic quality of the measure within the framework still needs some conceptual work.

10.1.2 Using the Zachman Framework to Measure Model Completeness (Semantic)

The Zachman framework, like the measures above, has been inspired by the architecture principles of building and construction engineering. The framework is described in more detail in

Chapter 3 as well as in [ZACH87 & SOWA92]. Of interest here is its potential application to model completeness analysis. Initially, Zachman developed it as an *information systems* framework, although it has subsequently been applied to the enterprise as a whole.

As mentioned in Chapter 3, the Zachman Framework has been applied in a number of different contexts. The following list gives some sample applications:

- To inform enterprise modelling methodologies [DEVI01].
- As the basis for building enterprise information architectures [COOK96].
- As a guide for the implementation of data warehouses [INMO97].

Because of its scope, it is hereby also suggested as a potential alternative semantic analysis technique to measure model completeness.

Conceptually, the framework suggests the major dimensions as well as the various levels of detail in which an organization can be modelled. In essence, therefore, the framework delineates the domain which should be modelled, and can thus be used as a yardstick to measure the content of the various enterprise models under consideration. The suggested procedure is to map all model entities (and possibly relationships and groupers) to the various cells of the framework. Where cells are relatively empty, this signifies a "gap" or lack of domain coverage.

An attempt at this analysis was made but a number of problems were experienced, some more practical and others more conceptual. The following are the main difficulties with adopting this type of framework-driven analysis:

- At first sight, the different columns of Zachman's framework appear to represent different, orthogonal dimensions. In fact, they are actually presented as such in the literature. After attempting (and failing) to allocate a number of entities to the various columns, it was found that they amount to different, complementary views or perspectives of the organization. For instance, the concept of a plan or budget involves time, people, motivation and process, and can therefore not be put in any one column. There are many such concepts in any of the enterprise models under consideration, for example even an apparently straightforward concept such as "employee" conflates some functional aspects with the people dimension.
- A difficulty of a different nature is experienced when attempting to place model elements in the appropriate row: it is often very difficult to determine the "correct" row for a given concept. The problem can be of a "spanning" nature (as above), whereby a concept straddles a number of (normally adjacent) rows such as the concept of plan, which can be at both the owner and the designer level. Most of the problems are more of a positioning nature, whereby it is unclear in precisely which row a concept belongs. Although the various levels appear quite distinct from a conceptual perspective, in practice the boundaries are continuous if not fuzzy, and the difference between for example a scope or owner view is sometimes hard to decide. Similarly, many concepts could easily be classified in either the builder view or the detailed view.
- It was therefore found that the allocation of model elements to the various cells is non-trivial and highly dependent on the researcher's interpretation of the concept and some of the model dimensions. This is not a desirable characteristic for an analysis method. Although a possible solution would be to assign a panel of experts to map each concept to a cell, it is not a practical approach in the light of the significant work involved (see below) and this would

still not resolve the above problems which would most likely be experienced by each individual expert.

- A related difficulty is the fact that the procedure cannot be automated. One of the objectives of the framework is to aim as much as possible for analysis methods (especially for the syntactic and semantic analysis) which can, in principle, be automated, whereas the above procedure is highly labour intensive.
- Even where cells could be filled with concepts, the difficulty arises on how to evaluate the final figures: should each cell contain a minimum number of model elements or a relative fraction? Is it desirable to have a fairly even spread of concepts across the various columns? As one travels down the rows, it can be expected to have a relative increase in the number of concepts mapped. When should the numbers stop or reverse (also related to the specificity of the model)? How many concepts are required to represent a given cell adequately? As an example, the "Data" column fills very quickly for most models, whereas the "Time" column is typically fairly sparsely populated. Is this natural? Necessary? Bad? Note that the type of model (e.g. an enterprise data model versus a process model or a use-case model) severely affects the number of entities allocated to each column; in fact the original suggestion by [DEVI01] is that a data model belongs entirely in the "Data" column.

In conclusion, the usefulness of the framework in evaluating enterprise models is not disputed. However, it is suggested that the Zachman framework is more useful as a reference when constructing a model rather than measuring the completeness of the model ex post. The strength of the model is in its structure, suggesting different perspectives to be addressed in the model. It would be difficult, if not inappropriate, to use the framework for model content analysis.

Note that the above comments can be applied, *mutatis mutandis*, to using the more recent Everington's Information Framework or IFW [EVER96] which has a total of 9 x 5 = 45 cells that need to be considered.

10.2 Building a "Consensual" or "Common Denominator" Generic Enterprise Model

One of the semantic analysis criteria (Chapter 8) was to determine the extent of overlap between the various models. One particular investigation was to look at the most central or important entities. The importance of an entity can be calculated by means of its fan-out i.e. the number of relationships in which it participates.

The following reiterates the 27 concepts (or direct synonyms thereof) that featured in the top 15 concepts list for at least 3 models:

> account; activity/process/action; address; agent/actor; asset;
> contract/agreement; corporation/business; cost/expense; customer/client;
> document; employee/human resource; environment;
> income/revenue/sales; income statement; inventory/stock; item;
> money/cash; order; organization unit; part; party; product; sale; supplier;
> time/period; transaction; unit.

These can therefore be considered to be *core* concepts for any enterprise model.

This suggests a methodology whereby a new generic enterprise model can be constructed that is a summary or consensus of the most important entities and relationships of a number of models.

10.2.1.1 Step 1: List common concepts

Firstly, the methodology as suggested in Chapter 8 can be used to generate those concepts that are covered by the most models i.e. the concepts which are included in the largest number of models. To reiterate in summary form, this involves splitting and normalizing all entity names into distinct word lists and using WordNet to generate lists of synonyms for all entity words. The following 53 concepts (or their synonym) were included in at least 18 (i.e. 80%) of the models:

Table 10-3: Most Common Model Concepts.

23 models	organization/organization
22 models	collection; product
21 models	activity; arrangement; component; content; part; production; state; system; unit
20 models	business; commodity; goods; issue; line; matter; merchandize; period; point; portion; position; substance; wares
19 models	account; beginning; concern; element; human; individual; information; message; person; program/programme; relation; somebody/someone
18 models	asset; bid; bidding; creation; grade; list; listing; mark; marketing; measure; merchandising; power; sale; selling; statement; way; work

Note that there is a significant overlap between the most common (Table 10-3) and the most important (Table 8-21)concepts, but there are also a number of differences. Some concepts are found in many models, but do not necessarily have a high fan-out. Typical examples are concepts that are generalizations (e.g. somebody/someone). The cut-off value for the minimum number of models in which a concept appears would be chosen by the researcher and determine the size of the resultant model. This obviously depends on the purpose of the model as well as on the quality of the models available in the database. A minimalist approach (as used above) will generate a very compact, easy to navigate, model but may lose too much detail to be of practical use apart from high-level pattern and template building.

A modification of the concept selection procedure might be used by introducing weights for each of the models. For example, one should consider that all concepts that appear in Ottawa-Dense, also appear in Ottawa-Big and Semi-Random and therefore automatically (but undeservedly?) receive a relative weight of "3". By giving each of the concepts in the Ottawa-based models a weight of, say $1/3^{rd}$, and by reducing the acceptance criterion to a required score of 16, this bias would be removed. Of course, it is easier just to remove the two derivative models Ottawa-Dense and Semi-Random from the database.

Conversely, concepts from models that are judged to be of high quality (e.g. SAP, Baan, Silverston) could receive a higher weight. They would then have a better chance of being included in the "common concepts list" than those from lower quality models (e.g. the random or semi-random model). The weighting could depend on the ultimate purpose of the model, e.g. different weights to models from the respective reference disciplines would be allocated, based on whether one wishes to build an ontology or a reference model for ERP systems.

10.2.1.2 Step 2: create synsets

The next suggested step is to conflate those concepts that are pure synonyms e.g. create the synsets corresponding to the original entities to which the terms referred:

"wares" = "goods" = "merchandise" = "commodity"; or

"human" = "individual" = "somebody/someone" = "person"

This is not quite as trivial as it seems, since the term "product", as found in 22 models, is representative of any item that is sold by a company (i.e. including services) whereas "goods/wares" may refer to material goods only. Similarly "organization" may have a different meaning than "business": "business" often refers to the enterprise being modelled whereas "organization" refers to a super-type which includes *any* party (actor) which is not an individual but can enter into agreements, including corporate customers and suppliers as well as the "business".

10.2.1.3 Step 3: add relationships and other model elements

Next, the relationships between the set of most common or popular entities must be constructed. There are again two potential approaches: a minimalist versus an inclusive approach. The inclusive approach includes all relationships in all the models between the selected entities, whereas the minimalist approach will require the relationship to occur in a researcher-specified minimum number of models e.g. the relationship must be found in, say, at least 10 of the models.

At this stage, additional model elements can be included such as entity attributes (again using a minimalist or inclusive approach), groupers and relationship attributes such as cardinalities. This depends on how fully specified the models in the database are, although the derivation of entity attributes does not necessarily require that all models in the database have attributes – using the "inclusive" approach, they could easily be derived from those models that include the necessary detail.

10.3 Extending the Framework to Other Research Areas in IS

Perhaps the most significant contribution of this research lies in the fact that the framework can, in principle, be applied to evaluate any set of intellectual works of a conceptual nature including literary works and works of art in any medium such as sculptures or paintings. More particularly, the following are areas within information science where the application of the suggested framework might be of use to structure evaluative and comparative research, including possibly the creative generation of new metrics.

- ***Model comparison***: it is obvious that models from other domains (i.e. not just enterprise models), can just as easily be compared using the proposed framework. For example, alternative information system models of the prototypical elevator application (as discussed in many systems engineering courses) could easily and comprehensively be compared by following an almost identical approach as was adopted in this research, substituting only the model database. Similarly, alternative models have been developed, and are available for, many other application areas such as real-time management systems for power stations or operating systems for telephone switch exchanges.
- ***Web site analysis***: this will be fully expanded in the section.
- ***User interfaces***: the user interface of different software packages (preferably within the same application category e.g. different operating systems or accounting packages) can easily be compared using the suggested framework. Note that the author is not very familiar with the very extensive body of research in this area, but consolidating and comparing the many approaches in this field by mapping their metrics into the suggested framework could be very interesting indeed.

- Typical ***syntax*** measures would be the number of screens and absolute/relative count of GUI elements (specificity/size), adherence to UI standards, consistency of screen layout (e.g. screen title or ID, how help is accessed, how dates are entered etc.), calculating a screen layout "temperature", applying the list of aesthetics metrics listed in Appendix C, computation of complexity indices (such as inter-linkages between screens, menu structure and individual screen composition).
- ***Semantic*** analysis would be concerned with the use of terms on screen by mapping them against lists of user vocabulary, comparing terminology between screens (consistency!) and across packages, use of synonyms, and readability of on-line help.
- ***Pragmatic*** measures would include screen response time (moving to the next screen, error messages, feedback for certain input), evolution of the screen layout between different versions (or platforms) of the same software, and the wide variety of statistics collected in usability labs such as Microsofts e.g. typical times taken by (novice versus expert) users to perform certain tasks.

- ***Documentation and training materials***: most information systems come with these; and there is often a choice of materials for a given information system. These could also be analysed using the framework. Some examples:
  - ***Syntactical*** measures would include size (in characters, words or pages), font size/types, number of graphics and tables, complexity (number of levels in the content hierarchy, number of cross-referenced terms e.g. by analysing the spread of the page numbers for index terms), number of errors, adherence to naming and layout standards, and the layout and temperature metrics mentioned earlier.
  - ***Semantic*** analysis could again look at the vocabulary used in the documentation, and map it not only against the language of the user (perspicuity), but also against the terms and language used in the (user interfaces of the) information system it is describing.
  - ***Pragmatic*** **analysis** would include cost, availability, reputation of supplier or author, and version (edition).
- ***Frameworks, architectures and methodologies:*** as explained in Chapter 3, information systems research has spawned a large number of methodologies (see methodology engineering!), frameworks and architectures. As a prime example of self-referential or meta-analysis, it is possible to use the proposed framework to categorize and compare (other) different IS frameworks, software and other architectures or system development methodologies. Applying this framework to the frameworks listed in [PERI93] or the methodologies surveyed in [JAYA94 or HUTT94] (and more recently developed ones) should prove to be straightforward yet illuminating.
- ***Algorithms:*** the framework can be used to compare different algorithms e.g. alternative sorting, compression or encryption algorithms in a given or several computer languages.
  - ***Syntactic*** measures are code size, compiled size, size of the data structures (given certain classes of problems), code efficiency, and complexity (using the traditional complexity metrics described earlier)
  - ***Semantic*** looks at the intrinsic meaning of the algorithm i.e. how does it function, how easily can the algorithm be understood (e.g. the quick sort is more difficult to grasp than

the binary or merge sort, which is more difficult to grasp than bubble and swap sort; a JPEG compression is conceptually more difficult than a GIF compression).

- ***Pragmatic*** analysis investigates the popularity of various algorithms, in what contexts are they used, how long it takes to code a given algorithm, who developed the algorithm (authority), how many refinements and optimizations have been developed, and how publicly available the algorithm is (e.g. for encryption, proprietary file formats or software protection schemes),

- ***Software applications:*** different software packages covering roughly the same application area (e.g. ERP systems, personal productivity suites, operating systems, statistical packages) can be compared using the framework.
  - Syntactic: code size, number of modules, number of files, complexity (of data files e.g. relational database structure, and application logic as discussed under algorithms), user interface analysis (as per above)
  - Semantic: looks at the functionality covered by each application both in breadth (number of different functions e.g. for statistical: how many different tests; for ERP: which functional areas) and depth (detail of implementation e.g. for statistical: how much and what feedback/output for each test; for ERP: how much of the functional area is implemented). Can also include semantic analysis of the user interface, documentation and training packages as described above.
  - Pragmatic: vendor support, documentation, price, required platform (hardware and operating system), reputation of vendor, upgrade/migration path.

- ***Programming languages:*** a lot of research has done in the area of comparative analysis of programming language. Since the syntax and semantics of programming languages are often described formally, and the distinction between them is made very explicit, it would be highly surprising if no systematic evaluation of programming languages along the lines suggested in the framework has been done. A literature search did not produce any prominent article using the framework fairly explicitly: many authors compare semantics and syntax explicitly, and give at the most a mere mention of pragmatics e.g. [MOSS01] Pragmatic aspects are important for the real world, and would include a systematic comparison, for instance, for various compilers available: their vendor, price, platform, efficiency and product maturity; but also the practical adoption of the various languages in the real world (typical applications, number of users), support structures (web communities and resources) etc.

- ***Software architectures:*** as mentioned in Chapter 5, it is interesting to note that [FABR98] came tantalizingly close to developing the framework by using the distinction between semantics, syntax and pragmatics in the contexts of assessing the quality of a software architecture, but they interpreted the terms quite differently and did not operationalize their approach in any practical way.

The list of potential application areas could go on, including e.g. ontology engineering, IT industry analysis, computational linguistics etc. Usually, a framework that has such a wide area of applications, can be accused of being so superficial or high-level, that no *practical* use or value can be derived from it. To counter this argument, the following gives a more in-depth and concrete research agenda based on the framework.

10.3.1 An Illustrative Application of the Framework to Website Analysis.

This section has been provided to illustrate the application of the framework in sufficient detail that it can form the concrete research agenda in an unrelated area: the analysis of websites. A typical example would be to compare, say, the websites of a number of competing companies within the same industry e.g. comparing the websites of different universities, on-line book sellers or search engines. The following proposed metrics and analyses should be regarded as suggestions, intended to illustrate how easily but fairly straightforward the various criteria can be mapped directly from this research to other information systems research area. Note that, just like for models, the framework does not accommodate composite web criteria such as overall usability or quality.

<u>Syntactic analysis</u>

- **Specificity (Size):** size of all information on the website, including a raw byte count (sub-grouped by text and multimedia elements), count of files (including graphics), pages, page widths etc. Could also include counts of tags (possibly according to categories), scripts, forms etc.
- **Correctness (Error Free):** adherence to standards (e.g. HTML 4.0, XML well-formedness etc), whether browser-specific tags are used, broken links etc.
- **Correctness (Consistency):** consistency of terminology across different pages (especially for large sites) as well as the formatting and layout style of web pages (including use of CSS etc.)
- **Architecture (Temperature):** Measuring the temperature of websites would be a novel and useful approach to website analysis, resulting in e.g. a web EQ (Emotional Quotient) index!
- **Architecture (Interface Layout/Aesthetics):** These metrics can easily be applied to webpage layouts, e.g. the home page of each website under investigation. Many of Nielsen's usability criteria would belong here too!
- **Complexity:** Measure the density of hyperlinks, both internal (within the web / could be relative links) and external (links to & from other websites). This includes all the formal complexity-based measures suggested by Thimbleby [THIM94].
- **Other syntactic metrics:** Includes a large number of other web-site metrics suggested in the e-commerce literature including, for example, some of Nielsen's usability criteria [NIEL99].

<u>Semantics</u>

- **Genericity: Coverage/Completeness:** Check each website to see what content they cover e.g. typically one would expect any reasonable standard organization's website to include the following (not necessarily as separate pages): welcome, contact details (with or without web enquiry form and/or e-mail contacts), organizational information (including structure and history), search facility, support, FAQ, product/service overview (including graphics), website feedback, ordering info/form/shopping cart, and other industry specific requirements.
- **Comparative Overlap:** Compute similarity indices between the different websites, either on a lexical basis (using words), or on a syntactical basis (web structure)
- **Perspicuity (lexicon):** Perform a readability analysis on the text of the website. Also map the vocabulary used in the website against the vocabulary of the intended audience.

Pragmatics

- **Validity (Authority):** Name brand recognition, corporate standing (including size, reputation etc.) and financial strength of the site owner
- **Validity (Use):** Web traffic statistics for each website. Note that, although exact statistics can normally be collected only by the hosting company and are normally confidential, it is possible get a rough estimate of these figures by analysing web-traffic through key-routers close to the hosting computer, as provided by some web-research companies.
- **Validity (Web links):** Detailed analysis of how often and by which method the site is referenced by other websites. Could also include the relative ranking of the websites in various search engines as well as banner analysis on other websites.
- **Purpose:** What e-commerce model is being used? How well does the web-site score on each of the levels in the e-commerce hierarchy (from passive information, ordering, process integration etc.)?
- **Cost:** How much did the website cost to construct? How much is spent maintaining the site (two proxy measures could be how many people are involved with the maintenance of the website and the size)?
- **Support:** How quickly does the web-administrator respond to feedback? How quickly does the organization respond to a webquery (e.g. enquiry about an order or submission of details using a web form)?
- **Other Pragmatic Criteria:** Hosting platform; browser compatibility; web-development tools used; web site responsiveness, empirical usability tests [BEVA94, NIEL99].

The above criteria can be compared to Zhu's suggested methodology which includes (only) 6 quality measures for web pages [ZHU00]. His quality measures include concrete metrics which are very similar to the ones used in this research:

- **Currency:** when last modified.
- **Availability**
- **Information to noise ratio**
- **Authority:** "The reputation of the organization that produced the web page", measured by a Yahoo Internet Review ranking (equivalent to the Google PageRank™).
- **Popularity:** Measured by how often cited (or referenced) by other web pages.
- **Cohesiveness**: Measured using a cosine similarity by mapping the number of concepts on a page against an 11-level 4385-node "topics" ontology.

10.3.2 When Can the Framework be applied to Other Research Areas?

Initially, the ease with which each of the criteria in the framework for model analysis can be mapped to the analysis of websites may appear surprising. At a more conceptual level, however, it can be argued that the website is also a conceptual reflection of its company i.e. it is a type of model representing the organization on the web.

This argument can be generalized by looking at the isomorphic features of the domain focus of a given research area on a meta-level. If sufficient isomorphic mappings exist between the main characteristics or attributes of the subjects being studied in two given research areas, then high-level analysis frameworks and techniques should prove to be fairly transferable. Note that it is this mapping that produces such powerful results in the field of mathematics, e.g. in group theory. The mapping requires a shift to thinking at a higher level of abstraction - a meta-level approach.

Table 10-4 illustrates some of these high-level correspondences between enterprise modelling and other research areas for which the framework is thought to be of use.

Table 10-4: Mapping the Isomorphism Between Some IS Research Areas.

A model	A website	An algorithm	A methodology	A software package
of a domain	for an organization	for a computable problem	for system development	for an application (user need)
is expressed in a modelling language	is coded in a markup language (e.g. HTML)	is coded in a language (or pseudo-language)	is expressed in a procedure manual or toolset	is coded in software instructions
and consists of a network of entities	and consists of a network of pages (concepts)	and consists of blocks of sequential computational steps	and consists of deliverables	and consists of dialogue screens
linked by relationships	linked by hyperlinks	linked by branches (loops, conditions and goto's)	produced by development processes	linked by user interactions (inputs values and icon clicks)

Note that Table 10-4 is intended mainly for illustrative purposes. Different mappings may be possible, depending on the type of analysis in which the researcher is interested. For instance, a software application could also be seen (conceptually) as consisting of collection of data sets, linked by the procedures (create, modify, delete) that operate on the data. For the analysis of scientific applications, this may well be the desired perspective, whereas the mapping in the table may be more appropriate for very interactive applications such as computer arcade games.

Finally, it is important to realise that these isomorphic mappings are not limited to information systems research only. A painting represents a subject by means of a composition of graphic elements placed in different positions on a tableau. A fiction book or play tells a story by means of situations whereby characters (and objects) are placed in a more or less complicated plot. A building is a physical construction consisting of various building blocks (walls, windows, and arches) organized in certain spatial relationships. A dictionary (or encyclopaedia) for a given language lists words in alphabetical order and explains them using other terms (found in the dictionary). These isomorphisms explain why some business lexicons could be viewed as enterprise models, why aesthetic analysis (originating from art) could be applied to GUI layout, or why information systems could make use of patterns and of the Zachman framework which has its roots in the built architecture and construction engineering.

Overall, the more tenuous the isomorphism, the more problematic the applicability of the framework (and the analysis techniques). Where the mapping appear to be more natural (e.g. between models and website), it is suggested that the transferability of the framework is accordingly more productive.

10.3.3 Applying the Framework to Evaluate Itself: A Self-referential Application.

The thesis that the framework can be applied to any "intellectual construct", including methodologies and frameworks, naturally leads to the question whether it can therefore be applied

to itself[6]. As explained in 4.2.1, self-referentiality it is not uncommon in IS as the examples of the UML (and meta-modelling in general) and the Zachman framework illustrate clearly.

To apply the framework to itself, all that is needed is to reorganize the appropriate validation criteria from section 5.4 and map them to the framework. This can be illustrated in the following table which summarises the arguments found in 5.4.

Table 10-5: Populated Proposed Framework for Model Analysis.

Syntactic	Semantic	Pragmatic
Conciseness: only three structuring concepts needed, Less than 20 criteria "clusters" cover the 80 criteria drawn from the literature (Appendix M). **Correctness**: not really applicable due to the lack of formality. **Integrity & Consistency:** fairly good due to objective criteria (repeatable) and unambiguous dimensions. **Modularity**: the framework has a natural but shallow structure. **Complexity:** the framework itself is not complex but some of the metrics to measure the criteria are. **Architectural style**: simplistic but clean.	**Genericity**: high as illustrated in section 10.3 **Completeness**: covers virtually all criteria suggested in a wide body of literature (see App. M). **Expressiveness:** the framework can accommodate any new criteria without ambiguity of location. **Similarity with other frameworks:** no current frameworks, but similar approaches advocated as discussed in 5.3.4 **Perspicuity/ Comprehens-ibility/ Understandability/ Self -descriptiveness:** the framework has an intuitive appeal and is easy to understand. **Documentation:** this thesis describes both the framework as well as its application at a great level of detail although the readability index is perhaps too high (i.e. dense reading!)	**Validity:** authority of the author rates fairly low (but builds on the work of respected academics) **Flexibility/ expandibility/ portability/ adaptability:** it is very easy to add / delete criteria **Purpose/ goal/ appropr-iateness/ relevance:** the initial purpose (evaluate enterprise models) is very clear. The applicability to other domains is the subject for future research. **Price/ cost:** free and public domain (academic). **Availability:** will be online (author and institutional website) as well as in university library. **Support:** it is the intention of the author to continue research in this area and it is the hope that other researchers will share in the research.

The arguments above are somewhat informal and subjective and intended more for illustrative purposes than as an actual rigorous scientific framework evaluation. Note that a number of validation criteria are not listed here but how trivially and naturally they can be accommodated within the framework structure. This exercise also illustrates how the framework can be used in a non-comparative way. The comparative equivalent – which will not be pursued further here –

[6] Actually, this was not such a natural and automatic suggestion. The credit for this idea must be (gratefully) attributed to an anonymous examiner of this thesis.

would be to compare the author's framework with the other frameworks which are reviewed in 5.2.

This final exploration of the conditions for applying the framework in other IS research areas, including itself, concludes the research. The next chapter is an attempt to provide a high-level *summary* of the results and contributions this research has made.

Chapter 11: Summary of Findings and Areas for Further Research

11.1 Review of the Research Objective

The original objective of this research was: "*the development and validation of a comprehensive framework for the analysis and evaluation of enterprise models*" (Chapter 1). It is believed that the objective has been achieved. This will be supported by a summary overview of the specific contributions and insights from this research which follows below.

The scientific contribution of this thesis can be summarized under the following headings:

- The enterprise model database.
- The detailed evaluation measures and metrics.
- The comparison of the enterprise models in the sample.
- The overall evaluative framework.

11.2 The Database of Enterprise Models

In order to validate and operationalize the framework, a test bed of more than twenty enterprise models was collected. Chapter 4 details the various pragmatic and theoretical criteria that were used for selecting models, which included a minimum size (more than 100 model elements), public availability, a wide variety of reference disciplines, some variation in quality and diversity of modelling paradigms. The aim was to be as inclusive as possible within the confines of these criteria, so an extensive search was performed to include as many enterprise models as possible.

A brief description of the more than twenty models which met the criteria can be found in Chapter 6, with more detailed information in Appendix A. Chapter 6 also describes nearly twenty other enterprise models that did not meet the criteria for inclusion into the database.

Since the enterprise data models in the sample originate from different methodological backgrounds, and vary significantly in the amount of detail they contain, there was a need to standardize the type of data captured. A standard meta-model for the model sample was used (see Chapter 4) to capture the selected models. Appendix A includes examples of the capturing process, conversion notes and sample model fragments. The fidelity with which the captured models represent the original model varies depending on the quality of the source. Some methodological problems were experienced with the capture of the Baan, SanFran and Miller models, though most of the other models should be seen as fair representations of the author's original models. To assist with the validity testing of the framework, two low quality models had to be constructed: a random and a semi-random model.

Below is a brief overview of the enterprise models which were selected and captured to serve as the test bed for validating the metrics. They are organized according to reference discipline.

- **Enterprise Resource Planning**. The reference models underlying the SAP R/3 [SCHE98] and Baan IV [PERR98] integrated enterprise applications were captured.
- **Data Model Libraries**. The following generic data model libraries have been captured from their respective published books: Marshall's BOMA [MARS00] and Fowler's analysis

patterns [FOWL97], which both follow the object-oriented paradigm, as well as Hay [1996] and Silverston [1997; 2001] which both use entity-relationship diagrams.

- **Academic Reference Models**. These include ARRI's Small Integrated Manufacturing Enterprise Model in IDEF0 notation and Purdue's Reference Model for CIM [WILL91] in DFD notation.
- **Enterprise Ontologies**. Two specific enterprise ontologies were selected, namely the Enterprise Ontology developed by the AIAI in Edinburgh [USCH98] and TOVE from EIL in Toronto. In addition, a subset containing all organization and enterprise-related concepts of the CYC Upper Ontology was selected.
- **Data Warehousing**. The only model in this category is Inmon's set of high and mid-level data models.
- **Financial Models.** These include the Belgian legally prescribed accounting framework [REYN94], and a financial spreadsheet model developed to model business growth at the USB in Stellenbosch [VANB88]
- **Framework Derived Models**. These include AKMA's Generic DataFrame and IBM's San Francisco Framework, the predecessor of WebSphere.
- **Miscellaneous Models**. For diversity and testing purposes, some other models were included namely Miller's General Living Systems Model (as an example of systems theory), the NHS's Generic Health Care Management Class Model (an example of an industry-specific model), Nippon Steel's small DFD model for CIM [WILL91], Ottawa's hyper-linked Business Dictionary as an example of a semantic model (two versions: 'Big' with all linked business terms and 'Dense' with those terms that have at least two links with other terms). Also included are two versions of a randomly generated model, with relationships built from random pairs of entities: the 'pure' random model contains terms selected 'at random' from the Oxford Paperback Dictionary [OXFO79] i.e. random syntax and random semantics and the semi-random model with the same random syntax but using business-related terms.

The researching, capturing and conversion constituted a significant time investment and the database of ready-captured, fairly large, real-world enterprise models represents an exciting resource for researchers interested in modelling research. Within the limitations of copyright legislation, the database with captured models will be made available to other researchers. Because of the proprietary nature of the some of the models, the definitions of the individual model elements cannot be made publicly available, although the most important information (including all syntactic information) will be made available on the Web. The full dataset is included in electronic format with this thesis for personal research purposes only.

The question arose as to the best format or platform in which the models should be made available. After careful consideration, it was decided that the only real alternative to consider is the use of XML (Extensible Markup Language). XML is a platform-independent, non-proprietary, internet-oriented way of sharing a hierarchically structured textual database. It is very flexible, human-readable in principle and allows easy transformation into other data formats.

The mapping of the meta-model to the XML tags that were used for the EnterpriseModels database, can be found in Appendix H. 37 different XML tags were used. The entire database has a size of 8,5 megabytes and can be viewed with normal text editors or word processors, XML-

enabled browsers (the latest versions of most standard browsers will suffice) or dedicated XML editors. Screen shots of the XML database as seen in the above software are also contained in Appendix H. A subset containing only 3 of the smaller models is also available for researchers wishing to experiment with the XML database, since this improves response times dramatically. Note that XML files are notoriously bloated, partly due to their human-readable tags. The zipped version of the database (including the DTD and XSD schema files) is only half a megabyte and can easily be distributed using a diskette.

It is hoped that this database will stimulate further research into models. Already, interest was expressed by researchers who are looking at using the database in commercial data mining and data cleaning tools. Another investigation currently underway is to incorporate the database in a proprietary knowledge base management tool aimed at the business analyst market.

11.3 Operationalizing the Framework and Validation of Model Analysis Techniques

A detailed summary of all of the metrics that have been tested can be found at the end of each relevant chapter i.e. sections 7.8, 8.9 and 9.7. Table 11-1 presents a summary of the most valid metrics or measures for each of the respective framework criteria.

11.3.1 Summary of Syntactic Measures

The syntactic model analysis is concerned with the purely structural aspects of the model, regardless of the underlying meaning of the model and its elements. Most of the analysis techniques are derived from the hard sciences i.e. the disciplines of software engineering, computer science, graph theory and network analysis. This includes a large variety of standard syntactic metrics relating to size, grouping, layering, inheritance structure, and network visualization, as well as some less standard metrics such as interface aesthetics.

Size is best measured with CASE size (or concept count) or the suggested adjusted concept count, because they do not favour the shallow models above the more fully specified models. Model ***correctness*** is measured against the formal modelling notation used by the model. Models could be rated using a composite "correctness score" measuring the number of errors, the degree of inconsistency as well as the use of standards. Model modularity can be measured by counting diagrams and groupers. The inheritance metrics give a fair feeling for the shape of the inheritance tree, and the degree to which inheritance is used in a model, but the interpretation of the metrics for comparing models leaves much to be desired and the validity of most of the *design* metrics suggested in the literature could not readily be validated at the analysis level. A similar argument applied to the ***complexity*** metrics where only relative connectivity and average fan-out displayed some validity. A ***graph plotting package***, such as Pajek, helped visualize the connectedness of a model e.g. similarly sized models could be compared using random plots or the Fruchterman-Reingold minimum-energy algorithm.

Combining the approaches from the graph analysis and system engineering metrics led to the idea of plotting and comparing ***the frequency distributions of the fan-outs*** for each model. The frequency distributions are indeed quite distinctive and characteristic of the underlying model. Because the classic descriptive frequency distribution statistics, proved to be not very adequate due to extreme skewness and large number of outlier values, ***alternative metrics*** were proposed: an intuitive categorization into curve types; the harmonic mean; and overall bumpiness or

smoothness. Finally, an attempt to measure ***architectural*** structure by means of a composite ***aesthetics*** metric shows promise.

Table 11-1: Summary of Validated Framework Metrics and Measures.

Framework Criterion	Section	Metric / measures
Semantic		
Size	7.1	CASE (concept) count and adjusted CASE count
Correctness; error-free; integrity; consistency	7.2	Combination of subjective syntax error, consistency and standards level score
Modularity	7.3	Nr of groupers, group levels and diagrams
Structure; hierarchy	7.3	Use of multiple inheritance; mean inheritance depth, reuse ratio.
Complexity; density	7.4	Relative connectivity; average fan-out; plot of Fruchterman-Reingold (for similar-sized models); harmonic mean of fan-out; fan-out distribution (chart); fan-out model signature.
Architectural style	7.7	Layout aesthetics
Semantic		
Genericity	8.3	G2 (average G1 score stability)
Coverage	8.3	Domain coverage score; core concept coverage
Completeness	8.8	Ranking of absolute lexicon coverage
Efficiency; conciseness	8.8	Relative lexicon coverage
Expressiveness	8.4	Average expressiveness score
Similarity & overlap with other models	8.7	Plot of similarity coefficients; most similar neighbours; similarity dendogram; most important concepts.
Perspicuity; comprehensibility; understandability; readability	8.5	NRAWPC (normalized rank-adjusted weighted perspicuity count)
Documentation	8.6	Completeness, extensiveness, readability (Flesh Reading Ease score)
Pragmatic		
Validity: authority & user acceptance	9.1	Academic author citations
Flexibility; expandibility; adaptability	9.3	Composite flexibility score
Currency; maturity	9.3	Descriptive table & taxonomy
Purpose; goal; relevance; appropriateness	9.6	Descriptive table
Availability	9.2	Medium & status
Cost	9.2	Purchase cost
Support	9.5	Tool & vendor suport, user base

11.3.2 Summary of Semantic Analysis

The semantic analysis of models is concerned with the intrinsic *meaning* of the model i.e. its relationship and mapping to the underlying domain reality it represents. Because semantic analysis is concerned with the correspondence between the abstract model and the underlying real domain, its reference disciplines are mainly linguistics, ontology research and lexicography, with much of the analysis concentrating on similarity, correspondence and cluster analysis. Many of

Model ***availability*** can be checked using accessibility, ***cost*** of the models, current model availability, physical size, and whether the model is available in digital format. A composite ***flexibility*** measure can be used that incorporates the availability in digital format, tool support, customisability and reusability and implementation independence.

Model ***currency*** and ***maturity*** refers to how often the model is updated, what the update policy (including version control) is and when the last update or change occurred. A somewhat "whimsical" taxonomy of models is: toys, fossils, souvenirs, trophies and stars. The rating of model ***support*** includes the following dimensions: tool support, vendor support and user base. Support was found to be closely correlated to the model currency and maturity typology. A final pragmatic consideration concerns the ***alignment*** of the model with the context in which it is to be used: theoretical foundation of the model, its effect on business or ***ease of integration***, its reference discipline and the model's purpose or ***goal***.

11.4 Comparative Enterprise Model Evaluation

Although the conceptual development and validation of the framework and its constituent criteria are considered to be the most important research contribution of this thesis, it is interesting to see if the individual measures can be combined in order to provide a high-level summary comparison of the models in the database.

It is obviously possible to perform a detailed in-depth analysis for any individual model by reiterating the findings for each individual measure and analysis mentioned in chapters 7 to 9. These are of interest to the developers of any specific model, since appropriate steps can be taken to improve a model's rating by analysing and improving those aspects of the model for which the measures are problematic e.g. addressing the perspicuity by renaming model elements, improving the readability of the documentation etc. In the interest of economy, the detailed findings for all measures on an individual model-by-model basis are not reiterated here.

However, by way of illustration, a number of model attributes relating directly to the composite index of overall model quality was selected. For each of these attributes, what appears to be the most valid or representative measure in this research was selected. Each model is then ranked based on the scores for each of these measures. The Random, Semi-Random and Ottawa-Big models have been omitted because they were included in the model database for validity testing.

The following measures were selected: Size = Adjusted CASE size; Complexity = Harmonic mean of fan-out; Genericity = Average domain coverage for different types of organizations; Perspicuity = NRAWPC using BL; Document readability = Flesch-Reading Ease; Most similar model = Smallest cosine distance; Completeness = Number of entity synonyms found in BL; Authority = Google PageRank™ for the organization's website.

The model rankings can be found in Table 11-2, where models are grouped according to similarity in the dendogram. The final column gives an "Average Ranking": the individual rankings for the various measures were combined into an overall score representing that model's average rank for the combined measures. All the models were then again ranked according to this overall (average rank) score. From a methodological point of view, this is not necessarily a very valid process since the relative weighting of various criteria is ignored, differences in model rankings (positions) for specific criteria do not reflect constant differences in the underlying scores on the basis of which the models were ranked, and the various null values ("not applicable") skew the rankings for quite a few criteria and models. Nevertheless, in the absence

of normalized analysis scores, it is better than no measure at all, as long as its interpretation is done with the above warnings in mind and taken as tentative. It must be reiterated here that, as suggested in Chapter 5, the "overall quality" of a model is indeed dependent on most of the above criteria (and others included in the framework), but that the relative importance (weighting) of each criterion is dependent on the purpose of the evaluation or intended model use.

Table 11-2: Model Ranking for Selected Model Quality Attributes.

		Syntactic					Semantic							Pragmatic			
Reference Discipline	Model ID	Model Code	SIZE	CORRECTNESS	COMPLEXITY	LAYOUT AESTHETIC	GENERICITY	EXPRESSIVENESS	PERSPICUITY	DOC. COMPLETENESS	DOC. READABILITY	MOST SIMILAR MODEL	COMPLETENESS	AUTHORITY	FLEXIBILITY	MATURITY & SUPPORT	"Average" RANKING
Dict	Ott.-Dense	**OD**	10		3		14	20	17	2	10	SR	6	7	3	toy	**10**
DW	Inmon	**IN**	3	17	20	3	7	16	10	19		OB	2	7	15	trophy	**16**
Fin	USB	**US**	13	9	7		3	7	6	17	3	OD	16	13	14	souvenir	**12**
Ontol -ogy	TOVE	**TO**	5	9	5		11	2	18	15	1	CY	8	7	8	borderline	**6**
	CYC	**CY**	1	6	10		19	3	13	2	12	BA	1	3	3	borderline	**3**
	AIAI	**AI**	18	2	11		2	1	15	1	13	HA	17	3	12	souvenir	**8**
ERP	SAP	**SA**	7	1	6	6	16	7	1	7	16	BA	4	2	3	star	**1**
	Baan	**BA**	6	6	8		16	15	2	15	8	SA	3	3	3	star	**5**
Data Models & Patterns	Silverston	**SI**	4		14	9	12	6	3	7	4	HA	6	13	1	trophy	**4**
	Hay	**HA**	2	2	4	10	12	3	9	7	2	SI	5	13	8	trophy	**2**
	BOMA	**BO**	13	2	11	8	7	3	7	11	5	SA	11	18	3	trophy	**7**
	SanFran	**SF**	17	15	9	5	10	10	10	14	7	BA	10	1	1	trophy	**9**
	Fowler	**FO**	16	6	15	7	3	12	16	7	6	HA	15	3	8	trophy	**11**
	NHS	**NH**	8	9	17		19	11	14	11	11	FO	12	7	12	trophy	**17**
	AKMA	**AK**	15	9	16	4	5	9	12	2	9	SF	18	13	15	trophy	**14**
CIM	Purdue	**PU**	11	15	1	2	14	16	4	11	17	NI	13	7	17	fossil	**15**
	Nippon	**NI**	19	9	18		18	12	5	19		SA	9	7	20	fossil	**20**
	ARRI	**AR**	12	9	11	1	7	12	8	2	15	NI	14	13	18	fossil	**13**
	BelgAcc	**BE**	9	2	19		5	19	20	17		HA	20		8	star	**19**
	Miller	**MI**	20		2		1	18	19	2	14	SA	19		18	fossil	**18**

Absolute rankings

A first evaluation can be made by looking at the overall rankings of the models, regardless of reference discipline. The *top scoring* models are: (1) Scheer's SAP reference model, (2) Hay's data model, (3) CYC's sub-ontologies related to organizations and (4) Silverston's data models. From a subjective evaluation of these models, based on their study throughout this research, a strong case can be presented that they indeed represent the best overall models in the database. It is a particular interesting vindication (and validation) of the framework that these models originate from three fairly different reference disciplines, yet can be compared despite their fundamentally different modelling approaches and philosophies.

Similarly, the models ranked at the *bottom of the scale* are indeed the "worst" models from a qualitative perspective: Both Nippon and BelgAcc are mainly listings of entities with very few relationships; Miller is a powerful but tiny and ill-specified model far removed from the mainstream; NHS is a vertical model which is very specific to the health care industry and Inmon

appears to be an inconsistent and very shallow model. Again, it is interesting that the framework manages to identify problematic models, regardless of the underlying reference discipline.

The rankings for the large group of models ranked between fifth and fifteenth position are less significant. From a subjective viewpoint, it can indeed be confirmed that these models are indeed neither top nor bottom in terms of quality, but their relative positions are more difficult to ascertain.

Relative rankings

A second interpretation of the rankings can be done by comparing the relative positions of models within the same reference discipline.

Between the two ERP models, SAP fairly consistently beats or matches Baan across various criteria (except documentation readability) and also overall. It must be stressed that the Baan model here is not the original conceptual model but a model re-engineered from a description of its relational implementation. This guarantees a quality handicap so the results should not be read as reflecting on the actual ERP packages. Nevertheless, there is support for the contention that the SAP model has a better theoretical foundation as well as represents significantly more analysis effort.

Comparing the data models, it must be admitted that the quality difference between Silverston and Hay is a tough call. However, of the more "pattern-like" models, BOMA is definitely cleaner that SanFran, bearing in mind the methodological problems in capturing SanFran. The position of Fowler is very debatable and it may well have been penalized because of its highly conceptual nature.

The comparative scores for the ontology-based models correspond excellently with the amount of ontology engineering and analysis effort invested in each of the models. CYC represents by far the most effort, followed by TOVE and then by AIAI. However, although smaller, AIAI is a more homogenous and conceptually higher level model, which is perhaps not fully reflected in the score.

Among the CIM models, Nippon well deserves its low score. ARRI is a much cleaner, more correct and rounded model that Purdue, though it is not clear whether it is indeed a better representation of the enterprise domain. Although it appears that Purdue represents significantly more modelling effort than ARRI, this is not reflected in the combined score. It is interesting to note their big value differences for the various metrics, and the validity of an "average ranking" must be considered with serious scepticism in this case.

Overall, it can be concluded from the detailed analysis that the techniques for most of the model evaluation criteria produce scores with good face validity. Generally, the validity is higher for comparing models within the same reference discipline than across disciplines.

A combined overall model ranking, perhaps representing some composite quality index, also has considerable face validity, especially for the extreme cases at either end of the quality spectrum. For pair-wise model comparison, the validity appears restricted to models where the scores for the individual criteria are consistently lower or higher.

11.5 The Framework for Evaluating Enterprise Models

11.5.1 Overall Framework Validity

This research has demonstrated that the suggested overall framework is a highly productive and valid approach for evaluating enterprise models. As shown in Chapter 5, all of the simple criteria which have been suggested in the literature can easily be classified within the framework. There remain a number of composite criteria such as overall "quality" and "usability", exact definitions of which tend to be fairly nebulous anyway. Even these composite criteria can benefit from the framework since their constituent attributes can be analyzed in terms of the framework.

The most useful dimension within the framework is the separation of syntactic, semantic and pragmatic criteria or factors. This distinction formed the basis for structuring the model analysis techniques in this thesis and a chapter was devoted to each category. It is very interesting to note that each chapter has a very distinct tone, reflecting a specific paradigmatic approach. Chapter 7, dealing with syntactic analysis, has a heavy pure science and engineering flavour, drawing heavily from computer science metrics, systems engineering graph-theoretical and even architectural concepts. Chapter 8, concerned with semantic analysis, relies mainly on lexicography and computational linguistics, as well as more conceptual information sciences such as meta-analysis or information frameworks. Finally, the relatively short Chapter 9, dealing with the pragmatic analysis, focused on practical business or commerce issues such as support, pricing, purpose, organizational impact etc. The framework thus brings together the basic constituent reference disciplines of information systems.

A second dimension was proposed for the framework, from absolute towards relative measures. This dimension is of a much more nebulous and ill-defined nature and was not a major focus of this research. It can be regarded as a suggestion for future research or elaboration.

The framework was validated both from a theoretical and an empirical point of view. The theoretical validation of the framework (see 5.4) emphasized its simplicity, perspicuity, completeness, flexibility, extensibility, universality (see also 10.3) and its sound theoretical foundation. Chapters 7 to 9 performed an in-depth empirical validation of the framework using the validation criteria specified in 4.2.3, illustrating that the framework is very practical and useful from an operational point of view. Various measures were developed to operationalize the framework and the most valid ones are detailed in Table 11-1 which also presents the final version of the framework.

Apart from the value of the framework to *classify* existing criteria, the framework should also be seen as an ideal way for the *creative design or generation* of new criteria or measures. Indeed, many metrics suggested in this research were inspired by thinking about model evaluation along the dimensions identified by the framework. It is therefore suggested that the framework could, and should, be applied to other areas of information systems research.

11.5.2 Application to Other Information Systems Research Areas

As pointed out in Chapter 10, the framework can, in principle, be applied to evaluate any set of intellectual works of a conceptual nature including literary works and works of art in any medium such as sculptures or paintings.

More particularly, it was suggested that any area where a degree of isomorphic mapping between the research subject and enterprise modelling exists, might be amenable to the adoption of the

framework as well as some of the analysis techniques suggested. The following are research areas within information science where the application of the suggested framework might be of use to structure evaluative and comparative research, including possibly the creative generation of new metrics. Detailed suggestions for criteria and analysis techniques can be found in Chapter 10, from which the following summary notes are quoted:

- **Model comparison**: models *from other domains* (i.e. not just *enterprise* models), can just as easily be compared using the proposed framework. For example, alternative models of a real-time control system could easily and comprehensively be compared by following an approach almost identical to the one adopted in this research, substituting only the model database.
- **Web site analysis**: Chapter 10 illustrates the application of the framework in sufficient detail so that it can form the concrete research agenda for what initially appears to be a completely unrelated area: the analysis of websites. A typical example would be to compare, say, the websites of a number of competing companies within the same industry, by comparing the websites of, for instance, different universities, online book sellers or search engines.
- ***Documentation and training materials***: Where there is a choice of training materials, such as tutorials for MS-Office, these could also be analysed using the framework.
- ***Frameworks, architectures and methodologies.*** IS research has spawned a large number of methodologies (methodology engineering), frameworks and architectures. As a prime example of self-referential or meta-analysis, it is possible to use the proposed framework to categorize and compare other different IS frameworks, software and other architectures or system development methodologies. Applying this framework to the frameworks listed in [PERI93], the methodologies surveyed in [JAYA94] or [HUTT94], and more recently developed ones, should prove to be an interesting exercise.
- ***Algorithms:*** the framework can be used to compare different algorithms e.g. alternative sorting, compression or encryption algorithms.
- ***Software applications:*** different software packages covering roughly the same application area (e.g. ERP systems, personal productivity suites, operating systems, statistical packages) can be compared using the framework.
- ***Programming languages:*** a lot of research in the area of comparative analysis of programming language uses the distinction between language syntax and semantics in a formal way. The framework provides a more systematic way, suggesting specific analysis techniques, as well as including the pragmatic dimension.
- ***Software architectures:*** as noted in Chapter 5, [FABR98] came tantalizingly close to developing the framework by using the distinction between semantics, syntax and pragmatics in the contexts of assessing the quality of software architecture, albeit using a different interpretation for some of the terminology. However, their approach confirms the applicability of the framework to software architecture analysis.

Usually, a framework that has such a wide area of application can be accused of being so superficial or high-level that no *practical* use or value can be derived from it. To counter this argument, Chapter 10 gives a more in-depth and concrete research agenda for *website analysis* based on the framework.

The chapter also points out that, the greater the underlying conceptual isomorphism between the research subjects of the above research areas, the more applicable or natural the transferability of the framework as well as specific analysis techniques will be.

11.6 Areas for Future Research

The following are some suggestions for future research, based on the analysis techniques and framework proposed in this thesis.

11.6.1 Developing the Second Dimension of the Framework

A weakness of the framework is the *one-dimensional* emphasis on the distinction between syntactic, semantic and pragmatic aspects. A suggestion was made in Chapter 5 to provide a second, not entirely orthogonal, dimension which classified the analysis techniques against the degree of objectivity of the reference criteria or standards. Apart from the fact that this second dimension is fuzzier and more gradual, its validity was not explored further in this thesis. The investigation of its validity within the context of model evaluation, as well as its applicability to other domains, would be an interesting extension. Alternatively, researchers could propose *another classification scheme* which is fully orthogonal to the primary dimension.

11.6.2 Refinement of Analysis Techniques

In most cases, it was found that the suggested analysis techniques and scores exhibited considerable face validity. There remained a few criteria for which an unsatisfactory operationalization was evident. Future research may well concentrate on alternative approaches to quantify these criteria. An example is the problem of measuring domain coverage and genericity.

Even for the better validated analysis techniques, there may remain considerable scope for further refinement and sophistication of the techniques. Of particular concern is the lack of comparability of the scores between the various measures. It should prove fairly straightforward to suggest some means for *normalizing* the scores by means of the use of a consistent scale.

In addition, there remains the scope for the formulation of alternative or equivalent analysis techniques which would present different perspectives or interpretations of the various criteria (e.g. model architecture). A number of pragmatic criteria could also be added.

11.6.3 More Emphasis on Relationships and Groupers for the Various Analysis Techniques.

Future research should place more emphasis on including more model elements, such as the entity attributes, groupers and relationships, in the analysis techniques. Most of the techniques in this thesis focused on model entities and often ignored relationship and groupers. This was done mainly because of the fact that too many models in the database did not name or describe their relationships or grouper constructs. A pragmatic methodology decision was usually taken to restrict the analysis to measuring the overlap between model entities, but future research should try to incorporate these. Hopefully more enterprise models will become available with properly named and defined relationships, or the framework will be validated in domain areas that have models with the required expressiveness. It must be added that most techniques that were suggested here are, *in principle*, equally applicable to entities as to other model elements.

Equally important, it must be reiterated that the focus of this research was on static data models. Future research should look at the evaluation of dynamic models, such as process models, where there appears to be more diversity both in modelling techniques as well as model content.

11.6.4 Validation of Newly Suggested Analysis Techniques in Other Domain Areas.

The framework was developed specifically for the evaluation of models "in general". Apart from one criterion, the "genericity", all other analysis techniques should be equally applicable to models from *other domain areas* as they did to *enterprise* models. This does not imply that the techniques are necessarily equally valid; some of the pragmatic scores are likely to change substantially, and some of the visualization metrics may become less valid or applicable.

It would be of particular interest to validate the framework by using a set of models produced by professional or student modellers who have been given an identical domain specification or description, and who are using the same modelling tool. This would provide a much more controlled environment, though some of the power of the framework lies in its capability to compare models from very different backgrounds and contexts.

11.6.5 Applying the Framework to Other Research Areas

A fairly complete case for the applicability of the overall framework, its criteria and the techniques suggested to operationalize these criteria, has been proposed in Chapter 10. Overall, the most fundamental condition for the applicability appears to be a degree of isomorphism between the "target" research disciplines. For example, in the sense that a web site (pages or topics linked by hyperlinks conceptually representing an organization) can be seen as an equivalent expression of a model (entities linked by relationships modelling an organization), the various analysis techniques can be applied, *mutatis mutandis,* with almost equal success. Specific suggestions to this effect have been made in Chapter 10.

A large number of other IS research areas also deal with conceptual objects displaying isomorphic similarities to models: algorithms, programming languages, methodology frameworks etc. Depending on the degree of isomorphism, the framework will also be more or less applicable or useful. Finally, it was suggested that this applicability is not necessarily limited to the discipline of information systems, but that fields such as ontology research, the visual and literary arts or even construction engineering might benefit from a similar conceptual approach.

Chapter 12: References

[AGAR99] Agarwal, R.; Bruno, G. and Torchiano, M., "Modeling Complex Systems: Class Models and Instance Models." Proceedings of the *International Conference on Information Technology (CIT'99).*

[AGIR00] Agirre, E.; Ansa, O.; Hovy, E. and D. Martinez. "Enriching Very Large Ontologies Using the WWW." Proceedings of the *First Workshop on Ontology Learning OL'2000* held in conjunction with the Fourteenth European Conference on Artificial Intelligence ECAI'2000, Berlin, Germany, 25 Aug 2000.

[ALFE94] Alfeld, L.E. *The System Dynamics Modeling Methodology.* White Paper, Decision Dynamics. http://www.decisiondynamics.com, accessed 10-Jan-2003.

[ALFR96] Alfred, Charlie and Mellor, Stephen L. "Observations on the Role of Patterns in Object-Oriented Software Development." In [BOWM96], pp. 279-287.

[ALJL01] Aljlayl, Mohammed and Frieder, Ophir. "Effective Arabic-English Cross-Language Information Retrieval via Machine-Readable Dictionaries and Machine Translation. *Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management*, Atlanta, Georgia, USA, November 5-10, 2001.

[AMAN99] Amann, B. and Fundulaki, I. "Integrating Ontologies and Thesauri to Build RDF Schemas." In *ECDL-99: Research and Advanced Technologies for Digital Libraries,* Lecture Notes in Computer Science, Paris, France, Sep 1999, pp. 234-253.

[AMBU99] Ambur, Owen D. "Freedom's Just Another Word ... for Metadata: Knowledge Management and Discovery via DASL, Z39.50, X.500, and the DMA." Proceedings of the *Third IEEE META-DATA Conference*, Bethesda, Maryland, 6-7 Apr 1999.

[ANH96] Anh, Dao Nam and Moore, Richard. *Formal Modelling of Large Domains*. Technical Report 74, UNU/IIST (United Nations University/International Institute for Software Technology), P.O.Box 3058, Macau, June 1996. Revised Sep 1996.

[ANON99] Anon. "Advantages of Outsourcing" in *IT Week*, December 1999 as quoted in "Hot Topics", http://uk.sun.com/services/hotspot/skills/hots/hottopics_1.html, accessed 10-Jan-2003 .

[APPL97] Appleton, B. *Patterns and Software: Essential Concepts and Terminology.* 1997. http://www.enteract.com/ ~bradapp/docs/patterns-intro.html, accessed 10-Jan-2003.

[AVIS95] Avison, D.E. and Fitzgerald, G. *Information Systems Development: Methodologies, Techniques and Tools.* London, UK: McGraw Hill, 1995.

[BARB95] Barbacci, Mario R.; Klein, Mark H.; Longstaff, Thomas A. et al. *Quality Attributes.* Technical Report 95-TR-021, Software Engineering Institute, December 1995.

[BARB97] Barbacci, Mario R.; Klein, Mark H. And Weinstock, Charles B. *Principles for Evaluating the Quality Attributes of a Software Architecture.* Technical Report 96-TR-036, Software Engineering Institute, May 1997.

[BART98] Bartlett, P.G. "Do you Need XML? A Checklist..." *SGML/XML GCA Conference*, Paris, France, May 1998.

[BARW74] Barwise, K.J. "Axioms for Abstract Model Theory." *Analytic Mathematical Logic*, Vol. 7 (1974), pp.221-265.

[BASI96] Basili, V.R.; Briand, L.C. and Melo, W.L. "A Validation of Object-Oriented Design Metrics as Quality Indicators." *IEEE Transactions on Software Engineering*, Vol. 22 No. 10 (Oct 1996), pp. 751-761.

[BASI88] Basili, V. R. and Rombach, H.D. "The TAME Project: Towards Improvement-Oriented Software Environments." *IEEE Transactions on Software Engineering,* Vol.14 No. 6 (Nov 1988), p.758-773.

[BATE90] Bateman, John A. "Upper Modeling: A General Organization Of Knowledge For Natural Language Processing." Proceedings of the *International Language Generation Workshop*, Pittsburgh, Pennsylvania, Jun 1990.

[BAUE98] Bauer, Christian and Glasson, Bernard. "Extending the Concept of a Reference Model Across Industries." Proceedings of the *IFIP WG8.2 & WG8.6 Joint Working Conference on Information Systems: Current Issues and Future Changes*, Helsinki, Finland, 10-13 Dec 1998, pp.471-488.

[BECH00] Bechhofer, S.; Broekstra, J.; Decker, S. et al. *An informal description of Standard OIL and Instance OIL.* Technical Report. Available from http://www.ontoknowledge.org/oil/, accessed 10-Jan-2003 .

[BECK00] Becker, Scot A. "Perspective And Abstraction." *The Data Administration Newsletter* Issue 15, Dec 2000. http://www.tdan.com/i015hy02.htm, accessed 10-Jan-2003

[BECK98] Becker, Scot A. "Building A Better Data Model." *The Data Administration Newsletter* Issue 7, Dec 1998. http://www.tdan.com/i007fe04.htm, accessed 10-Jan-2003

[BEEC87] Beech, David. "Groundwork for an Object Database Model." In [SHRI87], pp.317-354.

[BEER85] Beer, Stafford. *Diagnosing the System for Organisations.* Chichester, UK: J. Wiley, 1985. Reprinted 1995.

[BEER99] Beer, Stafford. "On the Nature of Models: Let Us Now Praise Famous Men and Women, Too (from Warren McCulloch to Candace Pert)." *Informing Science*, Vol. 2 No. 3 (1999), pp. 69-82.

[BENJ98] Benjamins, V.R.; Fensel, D. and Gómez-Pérez, A. "Knowledge Management through Ontologies." Proceedings of the *Second International Conference Practical Aspects of Knowledge Management (PAKM 98)*, Basel, Switzerland, 29-30 Oct 1998.

[BENN00] Ben-Natan, Ron and Sasson, Ori. *IBM SanFrancisco: Developer's Guide.* New York: McGraw-Hill, 2000.

[BENY90] Benyon, D. *Information and Data Modelling.* Oxford, UK: Blackwell, 1990.

[BERG00] Bergholtz, M. and Johannesson, P. "Validating Conceptual Models Utilising Analysis Patterns As An Instrument For Explanation Generation." *Fifth International Conference on Applications of Natural Language to Information Systems*, Métais, E. (ed.), Berlin: Springer Verlag, 2000.

[BERN01] Berners-Lee, T.; Hendler, J. and Lassila, O. "The Semantic Web." *Scientific American*, (May 2001), pp.28-37.

[BERN94] Bernus, P.; Nemes, L. and Morris, R. "Possibilities and Limitations of Reusing Enterprise Models." Proceedings of the *Second IFAC/IFIP/IFORS Workshop on Intelligent Manufacturing Systems, IMS'94*, Kopacek, Peter (ed), Vienna, Austria, Jun 1994, pp. 11-16.

[BERN96a] Bernus, Peter and Nemes, Laszlo (eds.). *Modelling and Methodologies for Enterprise Integration.* London, UK: Chapman & Hall, 1996.

[BERN96b] Bernus, Peter; Nemes, Laszlo and Williams, Theodore J. (eds.). *Architectures for Enterprise Integration.* London, UK: Chapman & Hall, 1996.

[BERN96c] Bernus, Peter; Nemes, Laszlo and Morris, R. "The Meaning of an Enterprise Model." In [BERN96a], pp.183-200.

[BEVA94] Bevan, N. and Mcleod, M. "Usability measurement in context" in *Behaviour and Information Technology,* 1994, pp. 132-145.

[BEVA95] Bevan, Nigel. "Measuring usability as quality of use" in *Software Quality Journal,* Vol 4 (1995), pp. 115-150.

[BEZI98a] Bezivin, Jean. "Who's Afraid of Ontologies?" Proceedings of *OOPSLA 98*, Vancouver, Canada, 18-22 Oct 1998.

[BEZI98b] Bézivin, Jean and Lemesle, Richard. "The sBrowser: a Prototype Meta-Browser for Model Engineering." Proceedings of *OOPSLA 98*, Vancouver, Canada, 18-22 Oct 1998.

[BEZI99] Bézivin, Jean. "Meta-Model Technology: Concepts And Applications." *XML'99 Conference*, Philadelphia, Pennsylvania, Dec 1999.

[BISS00] Bisson, G.; Nedellec, C. and Canamero, D. "Designing Clustering Methods for Ontology Building - The Mo'K Workbench." Proceedings of the *First Workshop on Ontology Learning OL'2000* held in conjunction with the Fourteenth European Conference on Artificial Intelligence ECAI'2000, Berlin, Germany, 25 Aug 2000.

[BJØR84] Bjørn-Andersen, N. "Challenge to Certainty." *Beyond Productivity: Information Systems Development for Organisational Effectiveness,* Bemelmans, T.M.A. (ed.). Amsterdam: North-Holland, 1984.

[BÖHM76] Böhm, B.W.; Brown, J.R. and Lipow, M. "Quantitative Evaluation of Software Quality." *Proceedings* of the *Second International Conference on Software Engineering on International Conference on Software Engineering,* Oct 1976, pp. 592 - 605.

[BÖHM78] Böhm, B. et al. *Characteristics of Software Quality.* New York: Elsevier North-Holland, 1978.

[BOLA87] Boland, R.J. and Hirschheim, R.A. (eds.). *Critical Issues in Information Systems Research.* Chichester,UK: J.Wiley, 1987.

[BOOC94] Booch, Grady. *Object-Oriented Analysis and Design with Applications.* Redwood City, California: Benjamin/Cummings, 1994.

[BOOC95] Booch, Grady and Rumbaugh, James. *Unified Method for Object-Oriented Development. Documentation Set. Version 0.8.* Santa Clara, California: Rational Software Corporation, 1995.

[BOOC99] Booch, Grady; Rumbaugh, James and Jacobson, Ivar. *The Unified Modeling Language User Guide.* Reading, Massachusetts: Addison-Wesley, 1999.

[BOWE95] Bowen, Jonathan P. and Hinchey, Michael G. "Seven More Myths of Formal Methods." *IEEE Software*, Vol. 12 No. 3 (1995), pp. 34-41.

[BRAN86] Brancheau, J.C. and Wetherbe, J.C. "Information Architectures: Methods and Practice." *Information Processing and Management*, Vol. 22 No. 6 (1986), pp. 453-463.

[BRAZ98] Brazier, Frances M.T. and Wijngaards, Niek J.E. "A Purpose Driven Method for the Comparison of Modelling Frameworks." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[BRIA94] Briand, L.C.; Morasca, S. and Basili, V.R. *Defining and Validating High-Level Design Metrics. Technical* Report CS-TR-3301, Version 2, University of Maryland, 1994.

[BRIN96] Brinkkemper, Sjaak; Lyytinen, Kalle and Welke, Richard J. (eds.). "Method Engineering: Principles of Method Construction and Tool Support." Proceedings of the *IFIP TC8, WG8.118.2 Working Conference on Method Engineering*, Atlanta, Georgia, 26-28 Aug 1996. London, UK: Chapman & Hall, 1996.

[BRIT94] Brito, E.; Abreu, F. et al. "Candidate Metrics for Object Oriented Software within a Taxonomy *Framework.*" *Journal of Systems and Software,* Vol. 26 No. 1 (1994), pp. 87-96.

[BROD82] Brodie, Michael L.; Mylopoulos, John and Schmidt, Joachim W. (eds.). *On Conceptual Modelling.* New York: Springer-Verlag, 1982.

[BRUN97] Bruno, G.; Reyneri, C. and Torchiano, M. "Enterprise Integration: Operational Models Of Business Processes And Workflow System." *Workshop on Enterprise Integration - International Consensus*, ICEIMT 97 International Conference on Enterprise Integration Modeling Technology, Torino, Italy, 28-30 Oct 1997.

[BUBE00] Bubenko, Janis A. and Kirikova, Marite. *Worlds In Requirements Acquisition and Modelling.* Unpublished Paper, Department of Computer and System Science, Stockholm University, 2000.

[BUBE86] Bubenko, Janis A. "Information System Methodologies - A Research View." *Information Systems Design Methodologies: Improving the Practice.* Olle, T.W.; Sol, H.G. and Verrijn-Stuart, A.A. (eds.), Amsterdam: North Holland, 1986.

[BUDA01] Budanitsky, Alexander and Hirst, Graeme. "Semantic Distance in WordNet: An Experimental, Application-oriented Evaluation of Five Measures". *Proceedings of the Workshop on WordNet and Other Lexical Resources*, in the North American Chapter of the Association for Computational Linguistics (NAACL), Pittsburgh, PA, June 2001

[BUDI96] Budinsky, F.; Finnie, M.; Vlissides, J. and Yu, P. "Automatic Code Generation from Design Patterns." *IBM Systems Journa,l* Vol. 35 No. 2 (1996), pp.151-171.

[BUSC96] Buschmann, Frank; Meunier, Regine; Rohnert, Hans; Sommerlad, Peter and Stal, Michael. *Pattern-Oriented Software Architecture: A System of Patterns.* Chichester, UK: J. Wiley, 1996.

[BYTH95] Bytheway, A. *Information Modelling for Management.* Working Paper SWP 17/95, School of Management, Cranfield University, U.K., 18 Aug 1995.

[CAIR98] Cairó, Osvaldo. "The KAMET Methodology: Content, Usage and Knowledge Modeling." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[CAMP96] Campbell, L. J.; Halpin, T.A. and Proper, H.A. "Conceptual Schemas with Abstractions: Making Flat Conceptual Schemas More Comprehensible." *Data & Knowledge Engineering*, Vol. 20 No. 1 (1996), pp.39-85.

[CARD90] Cárdenas, Alfonso F. and McLeod, Dennis. *Research Foundations in Object-Oriented and Semantic Database Systems.* Englewood Cliffs, California: Prentice-Hall, 1990.

[CATC87] Catchpole, C.P. *Information Systems Design for the Community Health Service.* Ph D Thesis, Aston University, Birmingham, 1987.

[CHAN98] Chandrasekaran, B.; Josephson, J. R. and Benjamins, V. Richard. "The Ontology of Tasks and Methods." Proceedings of the *Eleventh Workshop on Knowledge Acquisition , Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[CHEC81] Checkland, P.B. "Towards a Systems-based Methodology for Real-World Problem Solving." In [OPEN81], pp.288-314.

[CHEC92] Checkland, Peter. "Information Systems and Systems Thinking: Time to Unite?" In [COTT92], pp, 353-364.

[CHEN00] Chen, Z. and Zhu, B. *Some Formal Analysis of the Rocchio's Similarity-based Relevance Feedback Algorithm.* Technical Report CS-00-22, Dept. of Computer Science, University of Texas-Pan American, Mar 2000.

[CHEN94] Chen-Burger, Yun-Heh. *KBST: A Support Tool for Business Modelling in BSDM.* MSc. Thesis, Department of Artificial Intelligence, University of Edinburgh, 1994.

[CHEN97] Chen, Chaomei. "Structuring and Visualising the WWW by Generalised Similarity Analysis." *Proceedings* of the *Eighth ACM Conference on Hypertext,* Apr 1997, pp. 177-186.

[CHEN98] Chen-Burger, Yun-Heh; Robertson, David and Stader, Jussi. "Formal Support for an Informal Business Modelling Method." Proceedings of the *Tenth International Conference on Software Engineering and Knowledge Engineering (SEKE'98)*, 18-20 Jun 1998.

[CHIO02] Chiorean, Dan; Carcu, Adrian; Pasca, Mihai et al. "UML Model Checking" in *INFORMATICA*, Volume XLVII, Number 1, 2002, pp. 71-88.

[CHU97] Chu, Pai-Cheng. "Actors and Scripts: Object-Oriented Simulation of Enterprise Systems." In *Journal of Systems Software*, Vol 37 (1997), pp. 187-199.

[CHUR68] Churchman, C. West. *The Systems Approach.* New York: Delta, 1968.

[CIOC00] Ciocoiu, M. and Nau, D. "Ontology-Based Semantics." Proceedings of the *Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR'2000)*, Breckenbridge, Colorado, 12-17 Apr 2000.

[CLAX00] Claxton, John C. and McDougall, Peter A." Measuring The Quality Of Models." *The Data Administration Newsletter* Issue 14, Oct 2000. http://www.tdan.com/i014ht03.htm, accessed 10-Jan-2003.

[CLEM02] Clements, Paul; Kazman, Rick and Klein, Mark. *Evaluating Software Architectures: Methods and Case Studies.* London, UK: Addison-Wesley, 2002.

[CLEM94] Clements, Paul. From Domain Models to Architectures. *Workshop on Software Architecture*, USC Center for Software Engineering, Los Angeles, 1994.

[CODD93] Codd, E.F., Codd, S.B. and Salley, C.T. *Providing OLAP to User-Analysists: An IT Mandate.* Unpublished paper, Arbor Software Company, 1993.

[COFF97] Coffman, Bryan S. "James Miller's Living Systems Theory: An Interpretation and Application of the Model to Weak Signal® Research by Collaborative Design and Group Genius™ Processes." *Journal of Transition Management*, (Winter 1997). http://www.mgtaylor.com/mgtaylor/jotm/winter97/millerls.htm, accessed 10-Jan-2003.

[COLE93] Coleman, Michael Karl. *Aesthetics-based Graph Layout for Human Consumption.* Masters Thesis, Dept Computer Science, University of California, Los Angeles, 1993.

[COLE96] Coleman, Michael Karl & Stott-Parker, D. "Aesthetics-based Graph Layout for Human Consumption" in *Software – Practice & Experience.* Vol 26 No.12, pp.1415--1438, December 1996.

[COLO95] Colon, Carols. "Signification of Icons in a Computer GUI." Mar 1995. http://php.indiana.edu/~ccolon/Semiotics/ccolon2.html, accessed 10-Jan-2003.

[COMP98] Compatangelo, E. and Rumolo, G. "An Engineering Framework For Domain Knowledge Modelling." *Information Modelling and Knowledge Bases IX*, Amsterdam: IOS Press, 1998, pp. 51-65.

[COOK96] Cook, Melissa A. *Building Enterprise Information Architectures: Reengineering Information Systems.* Upper Saddle River, New Jersey: Prentice-Hall, 1996.

[COPL96a] Coplien, James O. "Pattern Languages for Organization & Process." In [BOWM96], pp. 237-248.

[COPL96b] Coplien, James O. "Pattern Mining." In [BOWM96], pp. 265-277.

[COPL98a] Coplien, James O. "A Generative Development-Process Pattern Language." In [RISI98], pp.243-300.

[COPL98b] Coplien, James O. "Software Design Patterns: Common Questions and Answers." In [RISI98], pp.311-320.

[COTT92] Cotterman, W.W. and Senn, J.A. (eds.). *Challenges and Strategies for Research in Systems Development.* New York: J. Wiley, 1992.

[COUR00] Courtot, Twyla. "What To Look For In Packaged Data Models/Databases." *Meta Data Conference*, Arlington, Virginia, 19-23 Mar 2000.

[COVE01] Cover, Robin. *XML Metadata Interchange (XMI).* http://www.oasisopen.org/cover/xmi.html, accessed 10-Jan-2003.

[CRAN99] Cranefield, Stephen and Purvis, Martin. "UML as an Ontology Modelling Language." Proceedings of the *IJCAI-99 Workshop on Intelligent Information Integration* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 31 Jul 1999.

[CRAW97] Crawley, Stephen; Davis, Scott; Indulska, Jaga; McBride, Simon and Raymond, Kerry. "Metameta is better-better!" *Workshop on Distributed Applications and Interoperable Systems (DAIS)*, Cottbus, Germany, Sep 1997.

[CROC91] Crockett, H.D.; Guynes, J. and Slinkman, C.W. "Framework for Development of Conceptual Data Modelling Techniques." *Journal of Information and Software Technology*, Vol. 33 No.2 (Mar 91), pp.134-142.

[CURR98] Curran, Thomas and Keller, Gerhard. *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model.* Upper Saddle River, New Jersey: Prentice-Hall, 1998.

[DAEL94] Daellenbach, Hans G. *Systems and Decision Making. A Management Science Approach.* Chichester, UK: J. Wiley, 1994.

[DAST97] Dastani, Mehdi and Indurkhya, Bipin. "An Algebraic Approach to Similarity and Categorization." *Interdisciplinary Workshop On Similarity and Categorization*, Edinburgh, Scotland, 1997.

[DAVE90] Davey, B.A. and Priestley, H.A. *Introduction to Lattices and Order.* Cambridge, UK: *Cambridge* University Press, 1990.

[DAVE98] Davenport, Thomas H. "Putting the Enterprise into the Enterprise System" *Harvard Business Review*, (July/Aug 1998), pp.121-131.

[DAVI93] Davis, R.; Shrobe, H. and Szolovits, P. "What is a Knowledge Representation?" *AI Magazine,* Vol. 14 No. 1 (Spring 1993), pp.17-33.

[DAVI99] Davidson, Andrew; Fuchs, Matthew; Hedin, Mette et al. Schema for Object-Oriented XML 2.0. W3C Note 30 July 1999. http://www.w3.org/TR/NOTE-SOX, accessed 10-Jan-2003.

[DAWS96] Dawson, Sandra. *Analysing Organisations.* Basingstoke, UK: Macmillan, 1996.

[DECK00] Decker, Stefan; Melnik, Sergey; Van Harmelen, Frank; Fensel, Dieter; Klein, Michel; Broekstra, Jeen; Erdmann, Michael and Horrocks, Ian. "The Semantic Web: The Roles of XML and RDF." *IEEE Expert,* Vol.15 No. 3, Oct 2000.

[DECK97] Decker Stefan; Daniel, M.; Erdmann, M. and Studer, R. 1997. "An Enterprise Reference Scheme for Integrating Model Based Knowledge Engineering and Enterprise Modelling." *Knowledge Acquisition, Modeling and Management, 10th European Workshop, EKAW'97.* Plaza, Enric and Benjamins, Richard (eds). Lecture Notes in Artificial Intelligence. Berlin: Springer. 81-96.

[DEME82] De Mey, Marc. *The Cognitive Paradigm.* Dordrecht, The Netherlands: D. Reidel, 1982.

[DEVI01] de Villiers, D.J. *Using the Zachman Framework to Assess the Rational Unified Process.* The Rational Edge, Mar 2001. http://www.therationaledge.com/content/mar_01/t_zachman_dv.html, accessed 10-Jan-2003.

[DIMI00] Dimitrov, M. "XML Standards for Ontology Exchange." Proceedings of *OntoLex 2000: Ontologies and Lexical Knowledge Bases,* Sozopol, Bulgaria, 8-10 Sep 2000.

[DOVE96] Dove, Rick; Hartman, Sue and Benson, Steve. *An Agile Enterprise Reference Model with a Case Study of Remmele Engineering.* An Agility Forum Project AR96-04, Dec 1996.

[DUCH00] Duch, W. "Similarity Based Methods: A General Framework For Classification, Approximation And Association." *Control and Cybernetics* Vol. 29, No. 4 (2000).

[EDMO99] Edmonds, Bruce. *Syntactic Measures of Complexity.* Doctoral Thesis, Dept of Philosophy, University of Manchester, 1999.

[EELE98] Eeles, P. and Sims, O. *Building Business Objects*. New York: J. Wiley, 1998.

[EHRI99] Ehrig, H.; Orejas, F. and Padberg, J. *Relevance, Integration and Classification of Specification Formalisms and Formal Specification Techniques.* Technical Report No. 99-13, TU BERLIN, Fachbereich Informatik, Aug 1999.

[ERIK00] Eriksson, H.-E. and Penker, M. *Business Modeling with UML. Business Patterns at Work.* New York: J. Wiley, 2000.

[EVAN93] Evan, William M. *Organization Theory: Research and Design.* New York: MacMillan, 1993.

[EVER96] Evernden, R. "The Information Framework" *IBM Systems Journal*, Vol. 35 No. 1 (1996), pp.37-68.

[FABR98] Fabbrini, Fabrizio; Fusani, Mario & Gnesi, Stafania. "Quality Evaluation based on Architecture Analysis." Proceedings of the International Workshop on the Role of Software Architecture in Testing and Analysis (ROSATEA'98), Marsala, Italy, 30-Jun to 3-July 1998. Available at http://www.ics.uci.edu/~djr/rosatea/papers/fusani.pdf, accessed 10-Jan-2003.

[FADE94] Fadel, George Fadi; Fox, Mark S. and Gruninger, Michael. "A Generic Enterprise Resource Ontology." Proceedings of *WET ICE'94: Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Morgantown, West Virginia, Apr 1994.

[FARO97] Faro, Alberto and Giordano, Daniela. "Between Narration And Drama: Information Systems Modelling Revisited." Proceedings of the *European Conference in Information Systems (ECIS'97)*, pp.384-395.

[FENS01a] Fensel, Dieter (ed.). "Ontologies and Electronic Commerce." *IEEE Intelligent Systems*, (Jan/Feb 2001), pp.8-14.

[FENS01b] Fensel, D. *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce.* Berlin: Springer-Verlag, 2001.

[FENS99] Fensel, Dieter; Angele, Jürgen; Decker, Stefan; Erdmann, Michael; Schnurr, Hans-Peter; Staab, Steffen; Studer, Rudi and Witt, Andreas. "On2broker: Semantic-Based Access to Information Sources at the WWW." Proceedings of the *IJCAI-99 Workshop on Intelligent Information Integration* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 31 Jul 1999.

[FENT96] Fenton, Norman; Pfleeger, E. and Shari, Lawrence. *Software Metrics : A Practical And Rigorous Approach.* London, UK: Thomson, 1996.

[FERN99] Fernández López, M. "Overview of Methodologies For Building Ontologies." Proceedings *of* the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods (KRR5), Stockholm, Sweden, August 2, 1999.

[FERR95] Ferrie, John. "Business Processes - A Natural Approach." Talk given at Warwick University on 19 Sep 1995. http://bprc.warwick.ac.uk/forum1.html, accessed 10-Jan-2003.

[FINN98] Finneran, Tom. "Enterprise Architecture: What And Why?" *The Data Administration Newsletter* Issue 7, Dec 1998. http://www.tdan.com/i007ht03.htm, accessed 10-Jan-2003.

[FLES79] Rudolf Flesch. *How to Write Plain English: A Book for Lawyers and Consumers* London, UK: Harper & Row, 1979.

[FLOR97] Florijn, G.; Meijers, M. and van Winsen, P. "Tool Support for Object-Oriented Patterns." Proceedings of *ECOOP '97*, Berlin: Springer-Verlag, 1997, pp. 472-495.

[FLYN96] Flynn, Donal J. and Diaz, Olivia Fragoso. *Information Modelling. An International Perspective.* London, UK: Prentice-Hall, 1996.

[FONG01] Fong, Joseph; Pang, Francis and Bloor, Chris. *Converting Relational Database into XML Document.* Working Paper, City University of Hong Kong, 2001. Available at http://www.cs.cityu.edu.hk/~jfong/WEBH2001.doc, accessed 10-Jan-2003.

[FORB95] Forbes, Paul. "Strategic Thinking: A Role for Soft Systems Methodology." *Second Australasian Conference in Strategic Management*, La Trobe University, Australia, Apr 1995.

[FORR71] Forrester, Jay W. *World Dynamics*. Waltham, Massachusetts: Pegasus Communications, 1971.

[FOWL97] Fowler, Martin. *Analysis Patterns*. Reading, Massachusetts: Addison-Wesley, 1997.

[FOX93a] Fox, M.S.; Chionglo, J.F. and Fadel, F.G. "A Common Sense Model of an Enterprise." Proceedings of the *Second Industrial Engineering Research Conference*, Northcross, Georgia: Institute for Industrial Engineers, 1993, pp. 425-429.

[FOX93b] Fox, Mark S. and Gruninger, Michael. *Ontologies for Enterprise Integration*. Unpublished paper, Dept of Industrial Engineering, University of Toronto, 30 Nov 1993.

[FOX95] Fox, Mark S.; Barbuceanu, Mihai and Gruninger, Michael. *An Organisation Ontology for Enterprise Modelling. Preliminary Concepts for Linking Structure and Behaviour.* Unpublished paper, Dept. of Industrial Engineering, University of Toronto, 1995.

[FOX98] Fox M.S. and Gruninger M. "Enterprise Modelling." *The AI Magazine*,(Fall 1998), pp. 109-121.

[FRAN98a] Frank, Ulrich. *The MEMO Meta-Meta Model*. Arbeitsberichte des Instituts für Wirtschaftsinformatik (Universität Koblenz-Landau) No. 9, (Jul 1998).

[FRAN98b] Frank, Ulrich. *Evaluating Modelling Languages: Relevant Issues, Epistemological Challenges and a Preliminary Research Framework.* Arbeitsberichte des Instituts für Wirtschaftsinformatik (Universität Koblenz-Landau) No. 15, (Dec 1998).

[FRAN99] Frank, Ulrich. *MEMO: Visual Languages For Enterprise Modelling*. Arbeitsberichte des Instituts für Wirtschaftsinformatik (Universität Koblenz-Landau) No. 18, (Jun 1999).

[FUNE01] Funes, Pablo. *Evolution of Complexity in Real-World Domains*. Ph D. Dissertation, Dept. of Computer Science, Brandeis University, 2001.

[GALE96] Gale, Thornton and Eldred, James. *Getting Results with the Object-oriented Enterprise Model*. New York: Prentice-Hall, 1996.

[GALI97] Galitz W.O. *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques*. New York: John Wiley, 1997.

[GAMM94] Gamma, Erich; Helm, Richard; Johnson, Ralph and Vlissides, John. *Design Patterns: Elements of Reusable Object-Oriented Software*. New York: Addison-Wesley, 1994.

[GARL93] Garlan and Shaw. "An Introduction to Software Architecture" in *Advances in Software Engineering and Knowledge Engineering, vol. I*. New York: World Scientific Publishing Company, 1993.

[GEIS98a] Geisler, R.; Klar, M. and Pons, C. "Dimensions and Dichotomy in Metamodeling." Proceedings of the *Third BCS-FACS Northern Formal Methods Workshop*. Berlin: Springer-Verlag, September 1998.

[GEIS98b] Geisler, R. "Precise UML Semantics Through Formal Metamodeling." Proceedings of the *OOPSLA'98 Workshop on Formalizing UML. Why? How?* Andrade, L.; Moreira, A.; Deshpande, A. and Kent, S. (eds.) 1998.

[GERB96] Gerbé, Olivier; Guay, B. and Perron, M. "Using Conceptual Graphs For Methods Modeling." *Conceptual Structures: Knowledge Representation as Interlingua,* Lecture Notes in AI 1115, P. W. Eklund, G. Ellis, and Mann, G. (eds.), Berlin: Springer-Verlag, 1996, pp. 161-174.

[GILL97a] Gillies, Alan. *Software Quality: Theory and Management.* London, UK: Thomson, 1997.

[GILL97b] Gillman, Daniel W. "Building A Statistical Metadata Repository." Proceedings of the *Second IEEE Metadata Conference*, Silver Spring, Maryland, 16-17 Sep 1997.

[GLAS01] GL Associates. *Chart of Accounts Design: Your Financial Data Warehouse.* http://www.glassoc.com/COA.asp, accessed 10-Jan-2003.

[GODI97] Godin, Robert; Mili, Hafedh; Mineau, Guy W. et al. "Design of Class Hierarchies Based On Concept (Galois) Lattices." *Theory and Application of Object Systems*, Vol. 4 No.2 (1997) pp. 117-134.

[GOGI95] Gogic, G.; Kautz, H.A.; Papadimitriou, C. and Selman, B. "The Comparative Linguistics Of Knowledge Representation." Proceedings of the *Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995, pp. 862-869.

[GOME99] Gomez-Pérez, A. and Benjamins, V.R. "Overview of Knowledge Sharing and Reuse Components: Ontologies and Problem-Solving Methods." Proceedings of the *IJCAI-99 Workshop on Ontologies and Problem-Solving Methods* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 2 Aug 1999.

[GOOS00] Goosenaerts, R. "Industrial Semiosis: Founding the Deployment of the Ubiquitous Information Infrastructure." *Computers in Industry* Vol. 43 (2000), pp. 189-201.

[GORM98] Gorman, Michael. "Data Model Evaluation Workplan." *The Data Administration Newsletter* Issue 5, Jun 1998. http://www.tdan.com/i005fe05.htm, accessed 10-Jan-2003.

[GREE91] Green, T.R.G.; Petre, M. and Bellamy, R.K.E. "Comprehensibility of Visual and Textual Programs: The Test of Superlativism Against the "Match-Mismatch" Conjectures." Proceedings of *Empirical Studies of Programmers: Fourth Workshop*, 1991, pp. 121-146.

[GREE95] Greenwood, R.M.; Robertson, L.; Snowdon, R.A. and Warboys, B.C. "Active Models in Business." Proceedings of *Fifth Annual Conference on Business Information Technology, BIT `95,* Department of Business Information Technology, Manchester Metropolitan University, 1995.

[GRUB93a] Gruber, Thomas R. *Toward Principles for the Design of Ontologies Used for Knowledge Sharing.* Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University.

[GRUB93b] Gruber T.R. "A Translation Approach To Portable Ontologies." *Knowledge Acquisition*, Vol. 5 No. 2 (1993), pp.199-200.

[GRUN96a] Grundy, John C. and Venable, John R. "Towards An Integrated Environment For Method Engineering." In [BRIN96], pp. 45-62.

[GRUN96b] Gruninger, M. and Fox, M.S. "The Logic of Enterprise Modelling." In [BERN96a], pp.140-157.

[GRUN96c] Gruninger, Michael. *Designing and Evaluating Generic Ontologies.* Working Paper, Department. of Industrial Engineering, University of Toronto, Jul 1996.

[GUAR96] Guarino, N. "Understanding, Building, and Using Ontologies." Proceedings of the *Knowledge Acquisition Workshop - KAW'96*, Banff, Canada, 9-14 Nov 1996. http://ksi.cpsc.ucalgary.ca/KAW/KAW96/guarino/guarino.html, accessed 10-Jan-2003.

[GUAR97a] Guarino, N. "Understanding, Building and Using Ontologies. A Commentary to 'Using Explicit Ontologies in KBS Development'." by van Heijst, Schreiber, and Wielinga. *International Journal of Human and Computer Studies,* Vol. 46 No. 2/3 (1997), pp. 293-310.

[GUAR97b] Guarino, N. "Some Organizing Principles for a Unified Top-Level Ontology." Revised version of a paper presented at *AAAI 1997 Spring Symposium on Ontological Engineering* (LADSEB-CNR Int. Rep. 02/97).

[GUAR98] Guarino, N. "Formal Ontology in Information Systems" Proceedings of the *First International Conference FOIS'98*, Trento, Italy, 6-8 Jun 1998. Guarino, N. (ed.), Amsterdam: IOS Press.

[GUAR99a] Guarino, N. "Avoiding IS-A Overloading: The Role of Identity Conditions in Ontology Design." Proceedings of the *IJCAI-99 Workshop on Intelligent Information Integration* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 31 Jul 1999.

[GUAR99b] Guarino, N.; Masolo, C. and Vetere G. "OntoSeek: Content-Based Access to the Web." *IEEE Intelligent Systems* Vol. 14 No. 3 (May/June 1999), pp. 70-80.

[GUST96] Gustas, R.; Bubenko, J. Jr. and Wangler, B. "Goal Driven Enterprise Modelling: Bridging Pragmatic and Semantic Descriptions of Information Systems." *Information Modelling and Knowledge Bases VII*, Tanaka, Y.; Kangassalo, H.; Jaakola, H. and Yamamoto, A. (eds.), Amsterdam: IOS Press, 1996, pp. 73-91.

[HALL90] Hall, J.A. "Seven Myths of Formal Methods." *IEEE Software*, Vol. 7 No. 5 (Sep. 1990), pp.11-19.

[HALP01] Halpin, Terry. "Augmenting UML with Fact-Orientation." Workshop proceedings: *UML: A Critical Evaluation and Suggested Future, HICCS-34 Conference*, Maui, Hawaii, Jan 2001.

[HALP99] Halpin, Terry and Bloesch, Anthony. "Data Modeling in UML and ORM: A Comparison." *The Journal of Database Management*, Vol. 10 No. 4 (Oct-Dec 1999), pp. 4-13.

[HAMM90] Hammer, Michael and McLeod, Dennis. "Database Description with SDM: A Semantic Database Model." In [CARD90], pp. 34-69.

[HAUS92] Hausen, *Hans*-Ludwig. "A Software Assessment and Certification Advisor." Proceedings of the *1992 ACM annual conference on Communications*, Apr 1992, pp. 309-316.

[HAY00] Hay, David C. "Data Model Views." *The Data Administration Newsletter* Issue 12, Apr 2000. http://www.tdan.com/i012ht01.htm, accessed 10-Jan-2003.

[HAY96] Hay, David C. *Data Model Patterns*. London, UK: Dorset House, 1996.

[HAY98a] Hay, David C. *Advanced Data Model Patterns*. http://www.tdan.com/i005fe03.htm

[HAY98b] Hay, David C. "Patterns and the Zachman Framework." *The Data Administration Newsletter* Issue 4, March 1998. http://www.tdan.com/i004fe07.htm, accessed 10-Jan-2003.

[HAYE99] Hayes, Glenda and Daniels, Robert M., Jr. "The Next Desktop Revolution: End-user Metadata." Proceedings of the *Third IEEE META-DATA Conference*, Bethesda, Maryland, 6-7 Apr 1999.

[HEIN01] Heinrich, Larry. "Using Template Data Models To Build The Data Warehouse" *Informix Technotes* Volume 7 (1991), Issue 1. Available online at http://www.adrm.com/1_infoart.htm, Accessed 12-Jul-2002.

[HEND96] Henderson-Sellers, Brian. *Object-Oriented Metrics : Measures of Complexity*. Upper Saddle River, New Jersey: Prentice-Hall, 1996.

[HENS94] Henschel, R. and Bateman, J. "The Merged Upper Model: a Linguistic Ontology for German and English." Proceedings of *COLING '94*, Kyoto, Japan, Aug 1994.

[HEYL92] Heylighen, Francis and Joslyn, Cliff. "What is Systems Theory." *Principia Cybernetica Web*. Heylighen, F.; Joslyn, C. and Turchin, V. (eds.) Brussels: Principia Cybernetica, http://pespmc1.vub.ac.be/SYSTHEOR.htm, accessed 10-Jan-2003.

[HICK99] Hicks, David; Tochtermann, Klaus; Rose, Thomas and Eich, Stefan. "Using Meta-Data to Support Customization." Proceedings of the *Third IEEE META-DATA Conference*, Bethesda, Maryland, 6-7 Apr 1999.

[HIRS89] Hirschheim, Rudy and Klein, Heinz K. "Four Paradigms of Information Systems Development". *Communications of the ACM*, Vol. 32 (Oct 1989), pp.1199-1216.

[HIRS97] Hirschheim, R.; Iivari, J. and Klein, H.K. "A Comparison of Five Alternative Approaches to Information Systems." *AJIS*, Vol. 5 No. 1 (Sep 1997), pp. 3-28.

[HO99] Ho, Ho-Chun. "Seeking Science in Art: Meta-Level Modeling. In *DM Review*, March 1999, p. 2-6.

[HOLL98] Hollocks, B.W. "The Role of Models in Leveraging Information" Proceedings of the *Second International Conference Practical Aspects of Knowledge Management (PAKM 98),* Basel, Switzerland, 29-30 Oct 1998.

[HOMM98] Hommes, Bart-Jan. *Analysing the Quality of a Business Modelling Technique.* Unpublished Masters Thesis, Delft University of Technology,1998.

[HONK98] Honkela, Timo. *Self-organizing Maps in Natural Language Processing.* Doctoral Thesis, Helsinki University of Technology, 1998.

[HSU94] Hsu, Cheng; Cho, Jangha; Yee, Lester and Rattner, Lauire. "Core Information Model: A Practical Solution to Costly Integration Problems." *Computers and Industrial Engineering,* (Aug 1994).

[HUBE94] Huber, H. "The Problems Of Data Modeling In Software Practice." Proceedings of the *First Workshop Knowledge Representation Meets Databases (KRDB '94)*, Saarbrücken, Germany, 20-22 Sep 1994.

[HUC97] Huc, Claude; Levoir, Thierry and Nonon-Latapie, Michel. "Metadata : Models and Conceptual Limits." Proceedings of the *Second IEEE Metadata Conference*, Silver Spring, Maryland, 16-17 Sep 1997.

[HULL90] Hull, Richard and King, Roger. "A Tutorial on Semantic Database Modeling." In [CARD90], pp. 1-3.

[HUTT94] Hutt, Andrew T. (ed.) *Object Analysis and Design. Description of Methods.* New York: J. Wiley, 1994.

[HYAT96] Hyatt, L. "A Software Quality Model and Metrics for Identifying Project Risks and Assessing Software Quality" in *Proceedings of the 8th Annual Software Technology Conference*, Utah, April, 1996.

[IBRA02a] O. Ibrahimov, I. Sethi, and N. Dimitrova, "The Performance Analysis of a Chi-square Similarity Measure for Topic Related Clustering of Noisy Transcripts." In *Proc. of 16th International Conference on Pattern Recognition (ICPR-2002)*, Quebec, Canada, 11-15 August 2002, Vol. 4, pp. 285-288.

[IBRA02b] O. Ibrahimov, I. Sethi, and N. Dimitrova, "Novel Similarity Based Clustering Algorithm for Grouping Broadcast News." *Proc. of SPIE Conf. 'Data Mining and Knowledge Discovery: Theory, Tools, and Technology IV'*, Vol. 4730, April 1-4, 2002, Orlando, Florida, pp. 394-304.

[IEEE92] IEEE Standard 1061-1992. *Standard for a Software Quality Metrics Methodology.* New York: Institute of Electrical and Electronics Engineers, 1992.

[IEEE01] IEEE. *Archive Standards in the Software Engineering Standards Subscription.* http://standards.ieee.org/catalog/olis/arch_swe.html, accessed 10-Jan-2003.

[INMO97] Inmon, W.H.; Zachman, John A. and Geiger, Jonathan G. *Data Stores, Data Warehousing and the Zachman Framework: Managing Enterprise Knowledge.* New York: McGraw-Hill, 1997.

[INMO99] Inmon, William H. *Using the Generic Data Model.* Unpublished White Paper. http://www.billinmon.com/library/whiteprs/earlywp/ttgendm.pdf, accessed 10-Jan-2003.

[ISAA99] Isaacs, Jeffrey D. and Aslam, Javed A. *Investigating Measures for Pairwise Document Similarity.* PCS-TR99-357, 1999.

[ISO98] ISO. *Software Product Quality.* ISO/IEC , Technical Report 9126, 1998.

[JACK97] Jackson, Michele H. "Assessing the Structure of Communication on the World Wide Web." *Journal of Computer-Mediated Communication*, Vol. 3 No. 1 (Jun 1997).

[JACO92] Jacobson, Ivar. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Reading, Massachusetts: Addison-Wesley, 1992.

[JARV88] Jarvenpaa, S.L. and Dickson, G.W. "Graphics and Managerial Decision Making: Research Based Guidelines." *Communications of the ACM*, Vol. 31 No. 6 (Jun 1988), pp. 764-774.

[JAYA94] Jayaratna, Nimal. *Understanding and Evaluating Methodologies. NIMSAD: A Systemic Framework.* London, UK: McGraw-Hill, 1994.

[JEUS01] Jeusfeld, Manfred A. and de Moor, Aldo. "Concept Integration Precedes Enterprise *Integration*." Proceedings of the *Hawaii International Conference on System Sciences*, 3-6, Maui, Hawaii, Jan 2001.

[JICP96] JICPA Information Systems Committee. *General Requirements of Accounting Information Systems*, Study Report No.14, Japanese Institute of Certified Public Accountants, Oct 1996.

[JOHN01] Johnson, Ralph. *Accounts: A Framework for Business Transaction Processing.* http://st-www.cs.uiuc.edu/users/johnson/Accounts.html, accessed 10-Jan-2003.

[JOHN87] Johnson, C.K. and Johnson, R.K. "Readability." *School Science Review*, Vol. 68 (1987), pp. 565-568.

[JOHN92] Johnson, Ralph E. "Documenting Frameworks using Patterns." Proceedings of *OOPSLA '92*, Vancouver, October 1992. New York: ACM Press, pp. 63-72.

[JOHN98] Johnson, Ralph E. "Introduction to Patterns". In [RISI98], pp. 353-360.

[JONK99] Jonker, Willem; Nijenhuis, Wim; Damen, Zef and Verwijmeren, Martin. "Workflow Management Systems and Cross-Organisational Logistics." Proceedings of the *Workshop on Cross-Organisational Workflow Management and Co-ordination*, San Francisco, California, 22 Feb 1999.

[KAAS96] Kaasbøll, J.J. and Smørdal, O. "Human Work As Context For Development Of Object Oriented Modelling Techniques." In [BRIN96], pp.111-125.

[KAN95] Kan, Stephen H. *Metrics And Models In Software Quality Engineering.* Reading, *Massachusetts*: Addison-Wesley, 1995.

[KANE68] Kane, Edward J. *Economic Statistics and Econometrics. An Introduction to Quantitative Economics.* New York, US: Harper & Row, 1968.

[KAPL98] Kaplan, David; Krishnan, Ramayya; Padman, Rema and Peters, James. "Assessing Data Quality in Accounting Information Systems." *Communications of the ACM*, Vol. 41 Issue 2 (Feb 1998), pp. 72-78.

[KARL94] Karlgren, Jussi and Cutting, Douglass. "Recognizing Text Genres with Simple Metrics Using Discriminant Analysis." Proceedings of *COLING 94*, Kyoto, Japan, pp. 1071-1074.

[KELL98] Kelly, Steven and Rossi, Matti. "Evaluating Method Engineering Performance: An Error Classification and Preliminary Empirical Study." *Australasian Journal of Information Systems*, Vol. 6 No. 1 (Sep 1998), pp.39-47.

[KENT97] Kent, Stuart; Lano, Kevin; Bicarregui, Juan; Hamie, Ali and Howse, John. "Component Composition in Business and System Modeling." Proceedings of the *OOPSLA'97 Workshop on Object-oriented Behavioral Semantics*, Technische Universität München, TUM-I9737, Kilov, Haim and Rumpe, Bernhard and Simmonds, Ian (eds.), 1997, pp. 91-97.

[KERH97] Kerhervé, Brigitte and Gerbé, Olivier. "Models for Metadata or Metamodels for Data?" Proceedings of the *Second IEEE Metadata Conference*, Silver Spring, Maryland, 16-17 Sep 1997.

[KIM95] Kim, Henry M.; Fox, Mark S. and Gruninger M. *An Ontology of Quality for Enterprise Modelling.* Unpublished paper, Dept. of Industrial Engineering, University of Toronto, 24 Jan 1995.

[KIRI00] Kirikova, Marite. "Explanatory capability of enterprise models." In *Data & Knowledge Engineering*, Vol 33 (2000), pp. 119-136.

[KIT95] Kit, Edward. *Software Testing in the Real World: Improving the Process*. Harlow, UK: Addison-*Wesley*, 1995.

[KNUT98] Knutilla, A. et al. *Process Specification Language: Analysis of Existing Representations*. NISTIR 6133, National Institute of Standards and Technology, Gaithershurg, Maryland (1998).

[KOLE93] Kolewe, R., "Metrics in object-oriented design and programming" in *Software Development*, Oct-1993, pp53-62.

[KORS92] Korson, T. and McGregor, J.D. "Technical Criteria For The Specification And Evaluation Of Object-Oriented Libraries." *Software Engineering Journal* Vol. 7 No. 3 (1992), pp. 85-04.

[KOSA99] Kosanke, K. and Nell, J.G. "Standardisation in ISO for enterprise engineering and integration." In *Computers in Industry,* Vol 40 (1999), pp. 311-319.

[KREM98] Kremer, Rob. "Visual Languages for Knowledge Representation." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[KROG95] Krogstie, J.; Lindland, O.I. and Sindre, G. "Defining Quality Aspects for Conceptual Models." In [FALK95], pp. 216-231.

[KURT99] Kurt, Christopher. "XML and Enterprise Application Integration." *XML 99 GCA Conference*, Philadelphia, Pennsylvania, Dec 1999.

[LAKE94] Lake, A. and Cook, C. "Use of Factor Analysis To Develop Oop Software Complexity Metrics." Proceedings of the *Sixth Annual Oregon Workshop on Software Metrics*, Silver Falls, Oregon, 1994.

[LAND87] Land, F.F. and Kennedy-McGregor, M. "Information and Information Systems: Concepts and Perspectives." In [GALL87], pp. 63-91.

[LAPR92] Lapri, J.C. (ed.). *Dependable Computing and Fault-Tolerant Systems. Vol. 5, Dependability: Basic Concepts and Terminology in English, French, German, Italian, and Japanese.* New York: Springer-Verlag, 1992.

[LARE96] Laresgoiti, I.; Anjewierden, A.; Bernaras, A.; Corera, J.; Schreiber, A.T. and Wielinga, B.J. "Ontologies as Vehicles for Reuse: A Mini-Experiment." Proceedings of the *Tenth Workshop on Knowledge Acquisition for Knowledge-Based Systems*, Banff, Canada, Nov 1996.

[LARM98] Larman, Craig. *Applying UML and Patterns. An Introduction to Object-oriented Analysis and Design.* Upper Saddle River, New Jersey: Prentice-Hall, 1998.

[LEE97] Lee, Lillian. *Similarity-Based Approaches to Natural Language Processing*. PhD thesis, Harvard University, 1997. (also Technical Report TR-11-97). Lewis, David D. and. Knowles, Kimberly A. Threading Electronic Mail: A Preliminary Study. Information Processing and Management, Vol. 33 No. 2 (1997) pp. 209-217.

[LEED01] Leedy, Paul D. and Ormrod, Jeannie Ellis. *Practical Research Planning and Design.* (7th edition), Saddle River, New Jersey: Prentice-Hall, 2001

[LEME98] Lemesle, Richard. "Meta-modeling and Modularity : Comparison between MOF, CDIF a sNets Formalisms." Proceedings of *OOPSLA 98*, Vancouver, Canada 18-22 Oct 1998.

[LENA90] Lenat, D.B.; Guha, R.V.; Pittman, K.; Pratt, D. and Shepherd, M. "CYC: Toward Programs With Common Sense." *Communications Of The ACM*, Vol. 33 No. 8 (1990), pp. 31-49.

[LEVY02] Levy, Steven. "The World According to Google" in *Newsweek*, 16 Dec 2002, pp. 44-47.

[LIDD95] Liddle, Stephen W. and Embley, David W. "Unifying Modeling and Programming through an Active, Object-Oriented Model-Equivalent Programming Language." Proceedings of the *Fourteenth International Conference In Object-Oriented & Entity-Relationship Modelling (OOER '95),* Australia, 13-15 Dec 1995, New York: Springer-Verlag, 1995, pp.55-64.

[LILE96a] Liles, D. H.; Johnson, M. E. and Meade, L. "The Enterprise Engineering Discipline." *Fifth Industrial 7 Engineering Research Conference*, Minneapolis, Minnesota, 1996, pp. 479-484.

[LILE96b] Liles, D. H. and Presley, A. "Enterprise Modeling within an Enterprise Engineering Framework." *96 Winter Simulation Conference*, Coronado, California, Association for Computing Machinery, 1996, pp. 993-999.

[LIN96] Lin, Jinxin; Fox, Mark S. and Bilgic, Taner. *A Requirement Ontology for Engineering Design*. Unpublished paper, Dept. of Industrial Engineering, University of Toronto, 1 Aug 1996.

[LIN98] Lin, Dekang. "An Information-Theoretic Definition Of Similarity." Proceedings of the *Fifteenth International Conference on Machine Learning*. San Francisco, California: Morgan Kaufmann, 1998, pp. 296-304.

[LOCH98] Loch, Joseph. "Corporate Data Model for System Integration." *Enterprise Strategies '98 COOL-BIZ Worldwide Customer Conference*, Dallas, Texas, 27 April-1May 1998. www.connect-usergroup.org/s98/add/ad158.pdf, accessed 10-Jan-2003.

[LONG98] Long, Kathy. "The Enterprise Data Model: A Key Ingredient For Successful Data Warehousing." *The Data Administration Newsletter,* Vol. 5 (Jun 1998). http://www.tdan.com/i005fe12.htm, accessed 10-Jan-2003.

[LOPE99] Lopez, M. Fernandez. "Overview of Methodologies for Building Ontologies." Proceedings of the *IJCAI-99 Workshop on Ontologies and Problem-Solving Methods* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 2 Aug 1999.

[LORE94] Lorentz, Mark and Kidd, Jeff. *Object-Oriented Software Metrics: A Practical Guide*. Englewood Cliffs, New Jersey: Prentice-Hall, 1994.

[LOUC95] Loucopoulos, P. and Kavakli, E. "Enterprise Modelling and the Teleological Approach to Requirements Engineering." *International Journal of Intelligent and Cooperative Information Systems*, Vol. 4 No. 1 (1995), pp. 45-79.

[LUKO96] Lukose, Dickson. "MODEL-ECS: Executable Conceptual Modelling Language." Proceedings of the *Tenth Workshop on Knowledge Acquisition for Knowledge-Based Systems*, Banff, Canada, Nov 1996.

[LYON98] Lyons, Brian. *Partner Perspective: Quantifying Quality in Your OO Models in Rose Architect*, Oct 1998. http://www.rosearchitect.com/mag/archives/9810/f6.html, accessed 10-Jan-2003.

[LYYT87] Lyytinen, Kalle. "A Taxonomic Perspective of Information Systems Development: Theoretical Constructs and Recommendations." In [BOLA87], pp.3-41.

[LYYT99] Lyytinen, Kalle and Welke, Richard. "Guest Editorial: Special Issue on Meta-Modelling and Methodology Engineering" in *Information Systems,* Vol 24 (1999) No. 24, pp.67-69.

[MADN99] Madnick, Stuart E. "Metadata Jones and the Tower of Babel: The Challenge of Large-Scale Semantic Heterogeneity." Proceedings of the *Third IEEE META-DATA Conference*, Bethesda, Maryland, 6-7 Apr 1999.

[MAED01b] Maedche, A. and Staab, S. "Learning Ontologies for the Semantic Web." Proceedings of the *Second International Workshop on the Semantic Web - SemWeb'2001*, Hong Kong, China, 1 May 2001.

[MAED01a] Maedche, A. and Staab, S. *Comparing Ontologies – Similarity Measures and a Comparison Study.* Internal Report 408, Institute AIFB, Karlsruhe University, 2001.

[MAGE94] Magee, S. and Tripp, L. (eds.). *Software Engineering: Standards and Specifications: An Annotated Index and Directory*. Denver, Colorado: Global Professional Publications, 1994.

[MANG99] Mangisengi, O.; Tjoa, A M. and Wagner, R.R. "Metadata for Data Warehousing Using Extended Relational Models." Proceedings of the *Third IEEE META-DATA Conference*, Bethesda, Maryland, 6-7 Apr 1999.

[MANS96] Manson, Gordon; North, Siobhán and Alghamdi, Abdullah. "Core Objects Required for a Generic CASE Repository." In [BRIN96], pp. 186-190.

[MARS00] Marshall, Chris. *Enterprise Modelling with UML. Designing Successful Software Through Business Analysis.* Reading, Massachusetts: Addison-Wesley, 2000.

[MARS98] Marshall, Chris. "Organisation in a Chaotic World." Proceedings of *OOPSLA'98*, Vancouver, Canada, 18-22 Oct 1998.

[MART00] Martin, P. "Conventions and Notations For Knowledge Representation And Retrieval." Proceedings of *ICCS 2000, 8th International Conference on Conceptual Structures, LNAI 1867*, Berlin: Springer-Verlag, Aug 2000, pp. 41-54.

[MART85] Martin, J. and McClure, C. *DiagrammingTechniques for Analyists and Programmers*. Englewood Cliffs, New Jersey: Prentice-Hall, 1985.

[MART95] Martin, James, and Odell, James. *Object-Oriented Methods: A Foundation.* Englewood Cliffs, New Jersey: Prentice Hall, 1995.

[MART96] Marttiin, P.; Harmsen, F. and Rossi, M. "A Functional Framework for Evaluating Method Engineering Environments: The Case of Maestro II/Decamerone and MetaEdit+." In [BRIN96] pp. 65-79.

[MAUG98] Maughan, Glenn and Avotins, Jon. "A Meta-model for Object-oriented Reengineering and Metrics Collection." *Eiffel Liberty Journal,* Vol. 1 No. 4 (Feb 98).

[MCCA76] McCabe, Tom. A. "Complexity Measure." *IEEE Transactions on Software Engineering*. Vol. 2 No. 4 (Dec 1976), pp. 308-320.

[MCLE01] McLeod, Graham. "Beyond UML: Advanced Systems Delivery with Objects, Components, Patterns and Middleware." Cape Town, South Africa: Inspired Press, 2001.

[MCLE92] McLeod, Graham. "A Model for Representation, Integration and Management of Methods." *South African Computer Journal*, 1992.

[MCLE98] McLeod, Graham. "Method Points: Towards a Metric for Method Complexity." *Australasian Journal of Information Systems*, Vol. 6 No. 1 (Sep 1998), pp. 48-58.

[MEIS95] Meis, Eike; Ostermayer, Rainer and Gittinger, Armin. *Guidelines on Domain Ontology Building*. KACTUS-01-RPK-D007-v1.1, Report, Kactus Project, Department of Social Science Informatics, University of Amsterdam, 1995.

[MELT96] Melton, Austin (ed.) *Software Measurement*. London, UK: Thomson, 1996.

[MEND00] Mendelson, Heim. *ERP Overview.* Research report, Graduate School of Business, Stanford University, Stanford, California, Jan 2000. Available: http://www.gsb.stanford.edu/CEBC/pdfs/ERP_Overview.pdf, accessed 10-Jan-2003.

[MENZ96] Menzies, Tim and Goss, Simon. *Vague Models And Their Implications For The KBS Design Cycle*. http://ksi.cpsc.ucalgary.ca/KAW/KAW96/menzies2/directory.html, accessed 10-Jan-2003.

[MENZ98] Menzies, Tim; Cohen, Robert F. and Waugh, Sam. "Evaluating Conceptual Modeling Languages." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[METZ96] Metzger, F. "The Challenge of Capturing the Semantics of STEP Data Models Precisely." Proceedings of the *PDTAG/PAKM Workshop on Product Knowledge Sharing for Integrated Enterprises PAKM'96*, Basel, Switzerland, Oct 1996.

[MILL01] Miller, Joe; Jacot, Allen & Stern, John A. *J.D. Edwards OneWorld: The Complete Reference*. New York: McGraw-Hill, 2001.

[MILL78] Miller, James Grier. *Living Systems*. New York: McGraw-Hill, 1978.

[MILL90] Miller, James Grier and Miller, Jessie L. "Introduction: The Nature of Living Systems." *Behavioral Science*, Vol. 35 No 3 (1990), pp. 157-163.

[MILL95] Miller, John H. *Evolving Information Processing Organisations*. Working Paper 95-06-053, Santa Fe Institute, http://www .santafe.edu/sfi/publications/Abstracts/95-06-053abs.htm, accessed 10-Jan-2003.

[MØLL01] Møller, Anders & Schwartzback, Michael. Problems with DTD. http://www.brics.dk/~amoeller/XML/schemas/dtd-problems.html, accessed 10-Jan-2003.

[MONT00] Montes-y-Gómez, Manuel; Gelbukh, Alexander and López-López. Aurelio "Comparison of Conceptual Graphs." In MICAI 2000: Advances in Artificial Intelligence. Lecture Notes in Artificial Intelligence N 1793 edited by Cairo, O. ; Sucar, L.E. and Cantu, F.J., Berlin, Springer-Verlag, pp. 548-556, 2000.

[MONT98] Montero, Luis and Scott, Colin T. "Improving the Quality of Component Business Systems with Knowledge Engineering." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[MORA94] Morale, Stuart. *Content, Form and Methods for Ontologies*. Unpublished notes of the Enterprise Project Workshop, Nottingham, UK, May 1994. http://www.aiai.ed.ac.uk/~bat/ontology-may94.html, accessed 10-Jan-2003.

[MOSS01] Mosses, Peter D. The Varieties of Programming Language Semantics And Their Uses. *Lecture Notes in Computer Science*, Vol. 2244, pp.165-190. Berlin: Springer-Verlag, 2001

[MULL97] Mullins, Craig. "The Capability Maturity Model - from a Data Perspective." *The Data Administration Newsletter* Issue 3, Dec 1997. http://www.tdan.com/i003fe04.htm, accessed 10-Jan-2003.

[MYER90] Myers, B. A. "Taxonomies of Visual Programming and Visualization." *Journal of Visual Language and Computing*, Vol. 1 No. 1 (1990) pp. 97-123.

[NAUL98] Nault, B.R. and Storey, V.C. "Using Object Concepts To Match Knowledge Representation Techniques To Problem Types." in *Information & Management*, Vol. 34 (1998), pp.19-31.

[NCIT97] National Committee for Information Technology Standards (NCITS): Technical Committee H7. *Object Model Features Matrix*. Document No. X3H7-93-007v12b; 25 May 1997. http://www.objs.com/x3h7/h7home.htm, accessed 10-Jan-2003.

[NELL00] Nell, J.G. *Designing Standards To Improve Process Interoperability*. A TC184 SC5 WG1 working paper, Mar 2000, unpublished. http://www.mel.nist.gov/sc5wg1/wg1vision.htm, accessed 10-Jan-2003.

[NELL96] Nell, J.G. "Enterprise Representation: An Analysis of Standards Issues." In [BERN96a], pp. 56-87.

[NELL99] Nell, J.G. *Enterprise Representation: A Different Paradigm For Designing Process-Interoperability Standards*. A TC184 SC5 WG1 working paper, Mar 1999, unpublished. http://www.mel.nist.gov/sc5wg1/paradigm.htm, accessed 10-Jan-2003.

[NEWT96] Newton, Judith. "Application of Metadata Standards." Proceedings of the *First IEEE Metadata Conference*, Silver Spring, Maryland, 16-18 Apr 1996.

[NG01] Ng, Wee Keong and Lim, Ee Peng. "Standardization and Integration in Business-to-Business Electronic Commerce." *IEEE Intelligent Systems*, Jan/Feb 2001, pp.12-14.

[NGO00] Ngo, David Chek Ling, Teo, Lian Seng and Byrn, John G. A Mathematical Theory of Interface Aesthetics. Unpublished working paper, available http://www.mi.sanu.ac.yu/vismath/ngo/, accessed 7-Mar-2002.

[NGO01] Ngo, David Chek Ling and Byrn, John G. "Another Look at a Model for Evaluating Interface Aesthetics", International Journal of Applied Mathematics and Computer Science (European publisher), Vol. 11, No. 2, 2001.

[NGUY94] Nguyen, Minh N. and Conradi, Reidar. "Classification of Meta-processes and their Models." Proceedings of the *Third International Conference on Software Process*, Seattle, Washington, 10-11 Oct 1994, pp. 167-175.

[NICK00] Nickols, F. *The Accountability Scorecard A Stakeholder-based Approach to "Keeping Score"*. http://home.att.net/~nickols/scorecrd.htm, accessed 10-Jan-2003.

[NIEL99] Nielsen J. *Designing Web Usability*. Indiana: New Riders Publishing, 1999.

[NILA90] Nilakanta, Sree; Wemhoff, Rebecca and Prebhu, G. M. "Knowledge-Based Graph Theoretic Analysis Of Data Flow Diagrams: Integrating CASE Tools With Expert Systems." Proceedings of the *ACM SIGBDP Conference on Trends and Directions in Expert Systems*, Sep 1990, pp. 58-71.

[NISS95] Nissen, H. W. and Zemanek, G. V. "Knowledge Representation Concepts Supporting Business Process Analysis." Working Notes of the *Second Workshop Knowledge Representation Meets Databases (KRDB '95)*, Bielefeld, Germany, 11-12 Sep 1995.

[NIST97] National Institute of Standards and Technology. *Approval Of Withdrawal Of Thirty-Three Federal Information Processing Standards Publications*. Docket No. 960726208-7142-02. http://www.itl.nist.gov/fipspubs/33fipwd.htm, accessed 10-Jan-2003.

[NOBL98] Noble, James. "Classifying Relationships Between Object-Oriented Design Patterns." *Australian Software Engineering Conference (ASWEC)*, 1998.

[NOY00] Noy, N.F. and Musen, M. A. "PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment." Proceedings of the *Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, Austin, Texas, 2000, pp. 450-455.

[NOY01] Noy, N.F. and McGuinness, D.L. *Ontology Development 101: A Guide to Creating Your First Ontology*. SMI technical report SMI-2001-0880 (2001).

[ODEL96] Odell, James J. and Fowler, Martin. "From Analysis to Design Using Templates." In [BOWM96], pp. 31-66.

[OEI95] Oei, J.L.H. "A Meta Model Transformation Approach Towards Harmonisation In Information System Modelling." In [FALK95] ,pp. 107-127.

[OLLE93] Olle, William T. "Data Modelling and Conceptual Modelling: A Comparative Analysis of Functionality and Roles." *Australian Journal of Information Systems*, Vol. 1 No. 1 (Sep 93).

[OMG97] OMG. *Business Object DTF. Common Business Objects v 1.5*. Object Management Group, 4 Dec 1997. http://www.omg.org, accessed 10-Jan-2003.

[OMG99] OMG. *Unified Modeling Language Specification*. Version 1.3, Jun 1999. http://www.omg.org, accessed 10-Jan-2003.

[OPEN97] Open Engineering Inc. *Business Objects: Definition, Taxonomy and Abstraction.* Unpublished briefing to the OMG BODTF CBO Working Group, 3 Nov 1997.

[ORLI96] Orli, Rick; Blake, L.; Santos, F. and Ippilito, A. *Address Data Quality and Geocoding Standards*. Unpublished report. http://www.kismeta.com/Address.html, accessed 10-Jan-2003.

[ORLI99] Orli, Rick. "The Launch of KisMeta Validator - A Meta-data Standards Setting and Enforcement Tool." Proceedings of the *Third IEEE META-DATA Conference*, Bethesda, Maryland, 6-7 Apr 1999.

[OXFO79] *The Oxford Paperback Dictionary*. Oxford, UK: Oxford University Press, 1979.

[PAIG97] Paige., R.F. "A Meta-Method for Formal Method Integration." Proceedings of *Formal Methods Europe '97*, Lecture Notes in Computer Science, Fitzgerald, John; Jones, Cliff B. and Lucas, Peter (eds.), Berlin: Springer-Verlag, 1997, pp.473-494.

[PARP98] Parpola, Päivikki. "Seamless Development of Structured Knowledge Bases." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 1998.

[PARS99] Parssian, Amir; Sarkar, Sumit and Jacob, Varghese S. "Assessing Data Quality For Information Products." Proceeding of the *Twentieth International Conference on Information Systems,* Jan 1999, pp. 428-433.

[PAZZ98] Pazzi, Luca. "Three Points of view in the Characterization of Complex Entities" in *Formal Ontology in Information Systems,* edited by Guarino, N. Amsterdam: IOS Press, 1998.

[PELT00] Peltier, Mikaël ; Ziserman, Francois and Bézivin, Jean. "On Levels of Model Transformation." *XML GCA Conference 2000*, Paris, France, Jun 2000.

[PERI93] Periasamy, K.P. "The State and Status of Information Architecture: An Empirical Investigation." Proceedings of the *Fourteenth International Conference on Information Systems*, Orlando, Florida, 5-8 Dec 1993, pp.255-270.

[PERK96] Perkins, Alan. *Developing a Data Warehouse: the Enterprise Engineering Approach.* Unpublished Paper, 1996. http://www.ozemail.com.au/~visible/papers/dw.htm, accessed 10-Jan-2003.

[PERR98] Perreault, Yves and Vlasic, Tom. *Implementing Baan IV.* Indianapolis, Indiana: Que, 1998.

[PINT99] Pinto, H.S.; Gómez-Pérez, A. and Martins, J.P. 'Some Issues on Ontology Integration." Proceedings of the *IJCAI-99 Workshop on Ontologies and Problem-Solving Methods* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 2 Aug 1999.

[POSC94] Petrotechnical Open Software Corp. *POSC Epicentre Data Model.* Englewood Cliffs, New Jersey: Prentice-Hall, 1994.

[POWE96] Powel, Antony and Vickers, Andrew. "A Practical Strategy For The Evaluation Of Software Tools." In [BRIN96], pp.165-185.

[PREE98] Preece, Alun. "Building the Right System Right. Evaluating V&V Methods in Knowledge Engineering." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[PRES97a] Presley, Adrien R. "A Multi-View Enterprise Modeling Scheme." Proceedings of the *Sixth Industrial Engineering Research Conference*, Miami Beach, Florida, 1997, pp. 610-615.

[PRES97b] Presley, Adrien R. *A Representation Method to Support Enterprise Engineering.* Doctoral Thesis, University of Texas (Arlington), May 1997.

[PRES97c] Pressman, Roger S. *Software Engineering: A Practitioner's Approach.* New York: *McGraw*-Hill, 1997 (4th ed.).

[PRIC98a] Price, B; Boxal, L. and Walsh, S. *Characteristics of Model-based Reference Architectures*. Technical Report, Dept of Information Systems, University of Cape Town, 1998.

[PRIC98b] Price Waterhouse Global Technology Centre. *Technology Forecast: 1998*. Menlo Park, California, 1998.

[PRIN96] Prins, R. *Developing Business Objects*. London: McGraw-Hill, 1996.

[PROB01] Probert, Stephen K. "Modeling using IS Methodologies: Some Guidelines Based on Authenticity and Contemporary Epistemology." *Informing Science*, (Jun 2001), pp. 437-443.

[PURC01] Purchase, Helen C.; McGill, Matthew; Colpoys, Linda and Carrington, David. "Graph drawing aesthetics and the comprehension of UML class diagrams: an empirical study." In *Proceedings of the Australian Symposium on Information Visualisation,* Sydney, Australia, 2001.

[REIJ98] van Reijswoud, Victor E. and Mulder, Hans B.F. *Speech Act Based Communication and Information Modelling with DEMO*. Unpublished Working Paper, Delft University of Technology, 1998.

[REIN94] Reingruber, Michael and Gregory, William. *The Data Modeling Handbook: A Best-Practice Approach To Building Quality Data Models*. New York: J. Wiley, 1994.

[RESN95] Resnik, P. "Using Information Content To Evaluate Semantic Similarity In A Taxonomy." Proceedings of the *Fourteenth International Joint Conference on Artificial Intelligence*, Montreal, Canada, 1995, pp.448-453.

[REUS97] Reus, B. and Streicher, T. "General Synthetic Domain Theory--A Logical Approach." Proceedings of *CTCS '97*, Springer LNCS 1290, 1997, pp. 293-313.

[REYN94] Reyns, Carl; Jorissen, Ann and Vanneste, Jacques. *Inleiding tot Accountancy*. UFSIA Universitaire Reeks Economie, Antwerp, Belgium, 1994.

[RISI98] Rising, Linda. *The Patterns Handbook: Techniques, Strategies and Applications*. Cambridge, UK: Cambridge University Press, 1998.

[ROBI97] Robinson, Bruce. "Notes for a Critical Theory of Representation in Information Systems." Proceedings of the *European Conference in Information Systems (ECIS'97),* pp.1134-1140.

[ROLL96] Rolland, Colette and Naveen, Prakash. "A Proposal For Context-Specific Method Engineering." In [BRIN96], pp. 191-208.

[ROSE97] Rosenthal, Arnon; Sciore, Edward and Renner, Scott. "Toward Unified Metadata for the Department of Defense." Proceedings of the *Second IEEE Metadata Conference*, Silver Spring, Maryland, 16-17 Sep 1997.

[ROSS96] Rossi, M. and Brinkkemper, S. "Complexity Metrics for System Development Methods and Techniques." *Information Systems*, Vol. 21 (1996), pp. 209-227.

[RUDL96] Rudloff, D. "Terminological Reasoning and Conceptual Modeling for Datawarehouse." Proceedings of the *Third Workshop Knowledge Representation Meets Databases (KRDB-96),* Budapest, Hungary, 13 Aug 1996.

[RUMB91] Rumbaugh, James et al. *Object-Oriented Modeling and Design*. Englewood Cliffs, New Jersey: Prentice Hall, 1991.

[SADI97] Sadiq, Wasim and Orlowska, Maria E. "On Correctness Issues in Conceptual Modelling of Workflows." Proceedings of the *European Conference in Information Systems (ECIS'97)*, pp.943-964.

[SAEK95] Saeki, Motoshi. "Object-Oriented Meta Modelling." Proceedings of the *Fourteenth International Conference In Object-Oriented & Entity-Relationship Modelling (OOER '95),* Australia, 13-15 Dec 1995, New York: Springer-Verlag, 1995, pp.250-259.

[SALI97] Salingaros, Nikos A. "Life and Complexity in Architecture From a Thermodynamic Analogy." *Physics Essays*, Vol. 10 (1997), pp 165-173.

[SALT93] Saltor, F. and García-Solaco, M. "Diversity with Cooperation in Database Schemata: Semantic Relativism." Proceedings of the *Fourteenth International Conference on Information Systems*, 5-8 Dec 1993, Orlando, Florida, pp. 247-254.

[SARG86] Sargent, Robert G. "The Use Of Graphical Models In Model Validation." Proceedings of the *Conference on Winter Simulation*, Dec 1986, pp. 237-241.

[SARR96] Sarris, A.K. "Towards Knowledge Representation: The State of Data and Processing Modelling Standards." *ISO Joint Workshop on Data and Process Modelling*, 9-12 Sep 1996, Bellevue, Washington.

[SCHE94] Scheer, August-Wilhelm. *CIM Towards the Factory of the Future*. Berlin: Springer-Verlag, 1994 (3rd ed.).

[SCHE98] Scheer, August-Wilhelm. *Business Process Engineering. Reference Models for Industrial Enterprises*. Berlin: Springer-Verlag, 1998 (2nd ed.).

[SCHE99] Scheer, August-Wilhelm. *ARIS Business Process Modeling*. Berlin: Springer-Verlag, 1999 (2nd ed).

[SCHL00] Schlenoff: C.; Gruninger M.; Tissot, F., Valois, J. et al. *The Process Specification Language (PSL): Overview and Version 1.0 Specification*, NISTIR 6459, National Institute of Standards and Technology, Gaithersburg, Maryland (2000).

[SCHL98] Schlenoff, C.; Ivester, R. and Knutilla, A. "A Robust Ontology for Manufacturing Systems Integration." Proceedings of the *Second International Conference on Engineering Design and Automation*, Maui, Hawaii, Aug 1998.

[SCHL99] Schlenoff, C.; Ciocoiu, M.; Lihes, D. and Gruninger, M. "Process Specification Language: Results of the First Pilot Implementation." Proceedings of the *International Mechanical Engineering Congress and Exposition*, Nashville, Tennessee, Nov 1999.

[SCHM00] Schmidt, Douglas; Stal, Michael; Rohnert, Hans and Buschmann, Frank. *Pattern-Oriented Software Architecture. Patterns for Concurrent and Networked Objects*. Vol. 2, Chichester, UK: J.Wiley, 2000.

[SCHR98] Schreiber, A.T. and Wielinga, B.J. "Knowledge Model Construction." Proceedings of the *Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, 18-23 Apr 1998.

[SEKA03] Sekaran, Uma. *Research Methods for Business. A Skill-Building Approach*. New York: J Wiley, 2003 (4th edition).

[SELI96] Seligman, Len and Rosenthal, Arnon. "A Metadata Resource to Promote Data Integration." Proceedings of the *First IEEE Metadata Conference*, Silver Spring, Maryland, 16-18 Apr 1996.

[SHAN97] Shanks, Graeme and Darke, Peta. "Using Explanation And Visualisation To Improve Understanding Of Corporate Data Models." *Eighth Australasian Conference on Information Systems*, Sutton DD (ed.), University of South Australia, 1997.

[SHEL96] Shelton, Robert E. "Business Objects: Modeling with Business Patterns." *Data Management Review*, May 1996.

[SHEP95] Shepperd, Martin. *Foundations of Software Measurement*. London, UK: Prentice-Hall, 1995.

[SILV97] Silverston, Len; Inmon W.H. and Graziano, Kent. *The Data Model Resource Book: A Library of Universal Data Models For All Enterprises*. New York: J. Wiley Computer Publishing, (1st ed.).

[SILV01] Silverston, Len; Inmon W.H. and Graziano, Kent. *The Data Model Resource Book: A Library of Universal Data Models For All Enterprises*, Revised Edition. New York: J. Wiley Computer Publishing, 2001.

[SILV98] Silverston, Len and Graziano, Kent. "Using Universal Data Models to Jump Start Your Data Modeling Effort." *Journal of Data Warehousing*, Data Warehousing Institute, Vol. 3, No. 1 (Spring 1998).

[SILV99] Silverston, Len. "Is Your Organization Too Unique To Use Universal Data Models?" *The Data Administration Newsletter* Issue 10, Sep 1999. http://www.tdan.com/i010fe04.htm, accessed 10-Jan-2003.

[SIMO99] Simons, Anthony J.H. and Graham, Ian. "30 Things That Go Wrong in Object Modelling with UML 1.3." *Behavioral Specifications of Businesses and Systems*, Kilov, H.; Rumpe, B. and Simmonds, I. (eds.), Kluwer, Dordrecht, The Netherlands, 1999, pp. 221-242.

[SIMS94] Sims, Oliver. Business Objects: Delivering Cooperative Objects for Client-Server. Boston, Massachusetts: McGraw-Hill, 1994.

[SINT01] Sintek, M.; Junker, M.; van Elst, L. and Abecker, A. "Using Information Extraction Rules for Extending Domain Ontologies." Proceedings of the *Second Workshop on Ontology Learning OL'2001* held in conjunction with the Seventeenth International Conference on Artificial Intelligence IJCAI'2001, Seattle, Washington, 4 Aug 2001.

[SKYT96] Skyttner, Lars. *General Systems Theory: An Introduction.* London: MacMillan, 1996.

[SMIT96] Smith, B.C. *On the Origin of Objects.* Cambridge, Massachusetts: MIT Press, 1996.

[SMIT98] Smith, Anne Marie. "Justification For An Enterprise Model." *The Data Administration Newsletter* Issue 4, Mar 1998. http://www.tdan.com/i004fe03.htm, accessed 10-Jan-2003.

[SNEL98] Snelting, G., and Tip, F. "Reengineering Class Hierarchies Using Concept Analysis." *Foundations of Software Engineering*, FSE-6, ACM, SIGSOFT Software Engineering Notes Vol. 23 No. 6 (1998), pp. 99-110.

[SOME99] Someya, Yasumasa. *A Corpus-based Study of Lexical and Grammatical Features of Written Business English.* Masters Dissertation, Graduate Dept of Language and Information Sciences, University of Tokyo, 1999.

[SOWA00] Sowa, John F. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Pacific Grove, California: Brooks Cole Publishing, 2000.

[SOWA92] Sowa, J.F. and Zachman, J.A. "Extending and Formalizing the Framework for Information Systems Architecture" *IBM Systems Business Journal*, Vol. 31 No. 3 (1992).

[STAA00] Staab, S. and Maedche, A. "Ontology Engineering Beyond the Modeling of Concepts and Relations." Proceedings of the *ECAI'2000 Workshop on Application of Ontologies and Problem-Solving Methods*. Amsterdam: IOS Press, 2000.

[STAD96] Stader, Jussi. "Results of the Enterprise Project." Proceedings of *Sixteenth Annual Conference of the British Computer Society Specialist Group on Expert Systems*, Dec 1996.

[STAM87] Stamper, R. "Semantics." *Critical Issues in Information Systems Research*, Boland, R.J. and Hirschheim, R.A. (eds.), Chichester: J. Wiley, 1987, pp. 43-78.

[STAM95] Stamper, R.K. "How Far Harmonisation? Information System Modelling Myopia." In [FALK95], pp.311-315.

[STEE00] Steen, M.W.A. and Derrick, J. "ODP Enterprise Viewpoint Specification." *Computer Standards and Interfaces.* Vol. 22 (2000), pp. 165-189.

[STER00] Sterman, John. *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Boston, Massachusetts: McGraw-Hill, 2000.

[STEV91] Stevenson, D.A. *Information Systems Architecture in the Large Systems Environment.* B.Com. (Hons) IS Technical Report, University of Cape Town, 1991.

[STEV01] Stevens, Perdita. Small-scale XMI programming: a revolution in UML tool use? Working paper, University of Edinburgh. http://www.dcs.ed.ac.uk/home/pxs/xmiImpact.pdf.

[STEW01] Stewart, Jim. "Enterprise-in-a-Box. An Enterprise Data Model Should Be The Cornerstone For Your Organization's It Development Efforts." *Intelligent Enterprise Magazine*, http://www.iemagazine.com/010723/411feat3_1.shtml, accessed 10-Jan-2003.

[STUD98] Studer, R.; Benjamins, V. and Fensel, D. "Knowledge Engineering: Principles and Methods." *IEEE Transactions on Data and Knowledge Engineering*, Vol. 25 (1998), pp. 161-197.

[STUR98] Sturlis, Paul. "Enterprise Information Architecture." *The Data Administration Newsletter* Issue 4, Mar 1998. http://www.tdan.com/i004fe10.htm, accessed 10-Jan-2003.

[SWAR96] Swartout, Bill; Patil, Ramesh; Knight, Kevin and Russ, Tom. "Toward Distributed Use of Large-Scale Ontologies." Proceedings of the *Tenth Workshop on Knowledge Acquisition for Knowledge-Based Systems,* Banff, Canada, Nov 1996.

[TAIL95] Tailor, D.A. *Business Engineering with Object Technology*. New York: J. Wiley, 1995.

[TAYI98] Tayi, Giri Kumar and Ballou, Donald. "Examining Data Quality." *Communications of the ACM*, Vol. 41, Issue 2 (Feb 1998), pp. 54-57.

[TERB97] ter Bekke, J.H. "Advantages of a Compact Semantic Meta Model." Proceedings of the *Second IEEE Metadata Conference*, Silver Spring, Maryland, 16-17 Sep 1997.

[TERH96] ter Hofstede, A.H.M. and Verhoef, T.F. "Meta-CASE: Is the Game Worth The Candle?" *Information Systems Journal*, No. 6 (1996), pp. 41-68.

[TERM01] Termier, A.; Sebag, M. and Rousset, M.-C. "Combining Statistics and Semantics for Word and Document Clustering." Proceedings of the *Second Workshop on Ontology Learning OL'2001* held in conjunction with the Seventeenth International Conference on Artificial Intelligence IJCAI'2001, Seattle, Washington, 4 Aug 2001.

[THEA01] Thearling, Kurt; Becker, Bany; DeCoste, Dennis; Malwby, Bill; Pilots, Michel and Sommerfield, Dan. "Visualizing Data Mining Models." *Information Visualization in Data Mining and Knowledge Discover.* Fayyad, Usama; Grinstein, Georges and Wierse, Andreas (eds.), Berlin: Morgan Kaufman, 2001.

[THIM94] Thimbleby, H. "Formulating usability" in *SIGCHI Bulletin*, April 1994, pp.59-64.

[TIP00] Tip, Frank and Sweeney, Peter F. "Class Hierarchy Specialization." *Acta Informatica* Vol. 36 No. 12 (2000), pp. 927-982.

[TODD98] Todd, Richard. *ERP: Time for a rethink.* White Paper, Max International, June 1998. Available from http://www.max-international.com, accessed 10-Jan-2003.

[TOIV01] Toivonen, S. Using RDF(S) to Provide Multiple Views Into A Single Ontology." Proceedings of the *Second International Workshop on the Semantic Web - SemWeb'2001*, Hong Kong, China, 1 May 2001.

[TURA02] Turau, Volker. DB2XML. A tool for transforming relational databases into XML documents. http://www.informatik.fh-wiesbaden.de/~turau/DB2XML/howto.html, accessed 10-Jan-2003.

[TYRR95] Tyrrell, A.J. *Eiffel Object-oriented Programming.* Basingstoke, UK: Macmillan Press, 1995. Also available on-line at: http://www.docm.mmu.ac.uk/library/notes/ajt/, accessed 10-Jan-2003.

[USCH96a] Uschold, Mike. "Converting an Informal Ontology into Ontolingua: Some Experiences." Proceedings of the *Workshop on Ontological Engineering held with ECAI 96*, Budapest, 1996.

[USCH96b] Uschold, Mike and Gruninger, Michael. "Ontologies: Principles, Methods and Applications." *Knowledge Engineering Review,* Vol. 11 No. 2 (Jun 1996).

[USCH98] Uschold, Mike; King, Martin; Moralee, Stuart and Zorgios, Yannis. "The Enterprise Ontology." *The Knowledge Engineering Review*, Vol. 13 (1998), special issue on Putting Ontologies to Use, Uschold, Mike and Tate, Austin (eds.).

[VALE96] Valente, Andre and Breuker, Joost. "Towards Principled Core Ontologies." Proceedings of the *Tenth Workshop on Knowledge Acquisition for Knowledge-Based Systems*, Banff, Canada, Nov 1996.

[VANB02] Van Belle, Jean-Paul. "Towards a Syntactic Signature for Domain Models: Proposed Descriptive Metrics for Visualizing the Entity Fan-out Frequency Distribution." *Proceedings of the SAICSIT 2002 Conference*, Port Elizabeth South Africa, Sep 2002, pp.19-29.

[VANB00] Van Belle, Jean-Paul. and Price, B. "A Proposed Framework for Evaluating Generic Enterprise Models." *South African Computer Journal*, Vol. 26 (2000), pp. 69-76.

[VANB88] Van Belle, Jean-Paul. *The USB Growth Model: A Multi-variable Financial Computer Model for Growth Businesses.* MBA Technical Report, University of Stellenbosch, 1988.

[VANB96] Van Belle, Jean-Paul. "A Critique of Current Systems Engineering Methods: The Case for Ontology-Augmented Methodologies." *Workshop on the Evaluation of Modelling Methods in Systems Analysis and Design*, Eighth Conference on Advanced Information Systems Engineering, Heraklion, Greece, 20-25 May 1996.

[VANB98] Van Belle, Jean-Paul. "Semantic Standardisation of High-level Enterprise Modelling. Arguments for an Ontological Approach." *Twenty Eighth Conference of the SA Computing Lecturers' Association*, Stellenbosch, South Africa, 28-30 Jun 1998.

[VANB99a] Van Belle, Jean-Paul and Eccles, Mike. *Discovering Information Systems.* Durbanville, South Africa: South African Universities Press, 1999.

[VANB99b] Van Belle, Jean-Paul. "Moving Towards Generic Enterprise Models: from Pacioli to Cyc." Proceedings of the *Tenth Australasian Conference on Information Systems (ACIS),* Wellington, New Zealand, Dec 1999, pp. 1084-1094.

[VAND99] van Deursen, Arie and Kuipers, Tobias. "Identifying Objects Using Cluster And Concept Analysis." Proceedings of the *Twenty First International Conference on Software Engineering,* 1999.

[VANH95] van Harmelen, Frank and Fensel, Deiter. "Formal Methods in Knowledge Engineering." *Knowledge Engineering Review*, Vol. 10 No. 4 (1995).

[VANH99] van Harmelen, Frank and Fensel, Deiter. "Practical Knowledge Representation for the Web." Proceedings of the *IJCAI-99 Workshop on Intelligent Information Integration* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 31 Jul 1999.

[VANS00] Van Scott. "The Last, Best Place to Integrate the Enterprise: Why Organizations Need an Enterprise Data Warehouse" in *The Data Administrator Newsletter*, Vol 12, April 2000. Available http://www.tdan.com/i013ht03.htm, accessed 10-Jan-2003.

[VENA93] Venable, John .R. *CoCoA: A Conceptual Data Modelling Approach for Complex Problem Domains.* Ph.D. dissertation, Thomas J. Watson School of Engineering and Applied Science, State University of New York at Binghamton, 1993.

[VERN97] Vernadat, F.B. "Enterprise Modelling Languages." Proceedings of *ICEIMT 97 International Conference on Enterprise Integration Modeling Technology*, Workshop on Enterprise Integration - International Consensus, Torino, Italy, 28-30 Oct 1997.

[VISS99] Visser, Pepijn and Tamma, Valentina. "An Experience with Ontology Clustering for Information Integration." Proceedings of the *IJCAI-99 Workshop on Intelligent Information Integration* held in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 31 Jul 1999.

[VONB81] Von Bertalanffy, Ludwig. "General System Theory - A Critical Review." In Open Systems Group (ed.). *Systems Behaviour.* London, UK: Harper-Row, 1981 (3rd ed.)., pp. 59-80.

[WALL01] Wallace, Dolores & Reeker, Larry: "Software Quality." In *SWEBOK Guide to the Software Engineering Body of Knowledge,* ISO/IEC JTC1/SC7 N2488 report, May 2001, pp.11-1 to 11-19.

[WAND90a] Wand, Yair and Weber, Ron. "An Ontological Model of an Information System." IEEE Transactions on Software Engineering, Vol. 16 No. 11(1990), pp. 1281-1291.

[WAND90b] Wand, Yair and Weber, Ron. "Mario Bunge's Ontology as a Formal Foundation for Information Systems Concepts." In: Weingartner, Paul and Dorn, Georg J.W. (eds.) *Studies on Mario Bunge's Treatise*. Amsterdam: Rodopi, 1990., pp. 123-150.

[WAND95] Wand, Yair and Weber, Ron. "On the Deep Structure of Information Systems." *Information Systems Journal*, Vol. 5 (1995), pp. 203-223.

[WASH01] The Washington Post Business Glossary. http://www.washingtonpost.com/wp-dyn/business/specials/glossary/index.html, accessed 10-Jan-2003.

[WATK98] Watkins, Bill. *Modelling the Firm as a Network.* Working paper 98-06-055E, Sante Fe Institute, 1998.

[WEBE99] Weber, Herbert; Klar, Marcus; Mann, Stefan; Kutsche, Ralf-Detlef et al. "Integrating Object-Oriented Modeling Techniques with Formal Specification Techniques. A Metamodel-based Approach." *Second German-Argentinian Workshop on Information Technology*, Königswinter, Germany, 3-5 Mar 1999.

[WEAT02] Weatherson, Brian. "Intrinsic versus Extrinsic Properties." *Stanford Encyclopedia of Philosophy*. http://plato.stanford.edu/entries/intrinsic-extrinsic/, accessed 10-Jan-2003.

[WHIT98a] White, J.C. *Building the Process-Based Organization*. White Paper, Decision Dynamics. http://www.decisiondynamics.com, accessed 10-Jan-2003.

[WHIT98b] White, J.C. and Alfeld, L.E. *Supply Chain Analysis Using System Dynamics Modeling and Simulation*. White Paper, Decision Dynamics. http://www.decisiondynamics.com, accessed 10-Jan-2003.

[WHIT97a] Whitman, L.E. and Huff, B.L. "A Living Enterprise Model" Proceedings of the *Sixth Industrial Engineering Research Conference*. Miami Beach, Florida, 1997, pp. 598-603.

[WHIT97b] Whitman, L.E.; Huff, B.L. and Presley, A.R. "Structured Models and Dynamic Systems Analysis: The integration of the IDEF0 and IDEF3 Modeling Methods and Discrete Event Simulation." Proceedings of the *Winter Simulation Conference*. Atlanta, Georgia, 1997, pp. 518-524.

[WHIT98c] Whitman, L.E.; Huff, B.L. and Presley, A. "Issues Encountered Between Model Views." *Flexible Automation and Integrated Manufacturing*. Portland, Oregon. 1998. pp. 117-130.

[WHIT98d] Whitman, L.E.; Huff, B.L. and Presley A. "The Needs And Issues Associated With Representing And Integrating Multiple Views Of The Enterprise." *Design of Information Infrastructure Systems for Manufacturing*, Ft. Worth, Texas, 1998, Mills, J. and Kimura, F. (eds.), IFIP, Amsterdam: Kluwer, 1998.

[WHIT99] White, Leonard. "Evaluating Measurement Methods." Proceedings of the *Nineth International Workshop of Software Measurement,* Magdeburg, Germany, 8-10 Sep 1999,.

[WILL91] Williams, T.J. (ed.). *A Reference Model For Computer Integrated Manufacturing (CIM). A Description from the Viewpoint of Industrial Automation.* CIM Reference Model Committee, The Instrument Society of America, Research Triangle Park, North Carolina, 1991 (2nd ed).

[WILL96] Williams, T.J. "The Needs of the Field of Integration." In [BERN96b], pp.21-31.

[WILL98] Williams, T.J and Li, H. "PERA and GERAM, Enterprise Reference Architectures in Enterprise Integration." *Design ofInformation Infrastructure Systems for Manufacturing*, Ft. Worth, Texas, 1998, Mills, J. and Kimura, F. (eds.), IFIP, Amsterdam: Kluwer, 1998.

[WILL02] Williams, Terry. *Modelling Complex Projects*. Chichester (UK): J. Wiley, 2002.

[WINC99] Winchester, G. (ed.) Industrial Automation Systems - Concepts and Rules for Enterprise Models. Proposed ISO standard WG1-N4 by TC184 SC5 WG1. Also referred to as ISO 14258:1998. Available at http://www.nist.gov/sc5wg1, accessed 10-Jan-2003.

[WITT92] Witt, B.I. "Software Architecture: Models and Methods." *Challenges and Strategies for Research in Systems Development*, Cotterman, W.W. and Senn, J.A. (eds.), Chichester, UK: J. Wiley, 1992, pp. 109-131.

[WITT94] Witt, B.I.; Baker, F.T. and Meritt, E.W. *Software Architecture and Design.* New York: Van Nostrand Reinhold, 1994.

[WORT00] Wortmann, J.C.; Hegge, H.M.H. and Rolefe, S. "Embedding Enterprise Software In Extended Enterprise Models." *Computers in Industry,* Vol. 42 (2000), pp. 231-243.

[WYNS96] Wyns, Jo; Van Brussel, Hendrik; Vackenaers, Paul and Bongaerts, Luc. "Workstation Architecture in Holonic Manufacturing Systems." Proceedings *Twenty Eighth CIRP International Seminar on Manufacturing Systems*, Johannesburg , South Africa, 15-17 May 1996, pp. 220-231.

[XU00] Xu, Hongjiang. "Managing Accounting Information Quality." Proceedings of the *Twenty First International Conference On Information Systems*, Dec 2000, pp. 628-634.

[YAMA01] Yamaguch, T. "Acquiring Conceptual Relationships from Domain-Specific Texts." Proceedings of the *Second Workshop on Ontology Learning OL'2001* held in conjunction with the Seventeenth International Conference on Artificial Intelligence IJCAI'2001, Seattle, Washington, 4 Aug 2001.

[YEO96] Yeo, Gee Kin; Wang, Shi Ping and Hu, Jian. "An Object-oriented Architecture for Model Representation." *IASTED International Conference on Modelling, Simulation and Optimization*, Gold Coast, Australia, May 1996. ftp://ftp.comp.nus.edu.sg/pub/staff/yeogk/oomr.doc, accessed 10-Jan-2003.

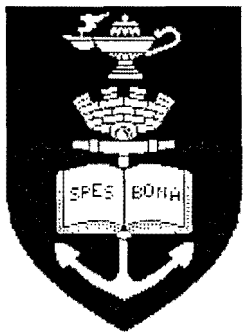
[ZACH87] Zachman, John. "A Framework for Information Systems Architecture." *IBM Systems Journa,l* Vol. 26 No. 3 (1987).

[ZACH97] Zachman, John A. "Enterprise Architecture: The Issue of the Century." *Database Programming and Design*, Mar 1997.

[ZHANG00] Zhang, Dell and Dong, Yisheng. "An Efficient Algorithm to Rank Web Resources." Proceedings of the *Nineth International World Wide Web Conference*, Amsterdam, 15-19 May 2000.

[ZHU00] Zhu, Xiaolan and Gauch, Susan. "Incorporating Quality Metrics In Centralized/Distributed Information Retrieval on the World Wide Web." Proceedings of the *Twenty Third Annual International ACM SIGIR Conference on Research and Development In Information Retrieval,* Jul 2000, pp. 288-295.

[ZUSE99] Zuse, Horst. "Validation of Software Measures and Prediction Models." Proceedings of the *Nineth International Workshop of Software Measurement,* Magdeburg, Germany, 8-10 Sep 1999.



Appendices

for

"A Framework

for the Analysis and Evaluation

of Enterprise Models"

Jean-Paul W. G. D. Van Belle

Thesis Submitted

for the Degree of Doctor of Philosophy

to the Department of Information Systems

University of Cape Town

February 2003

List of Appendices

APPENDIX A: MODEL SUMMARY DATA, SAMPLE MODEL FRAGMENTS, MODEL CAPTURE AND CONVERSION NOTES

This appendix includes examples of the capturing process, conversion notes and sample model fragments. In addition, the high level model summary attributes are included for each of the models. The model summaries included three groups of model attributes listed in this appendix.

- Model biography: identifying model characteristics including name and source.
- Summary metrics: some high-level metrics to indicate the main model statistics.
- Meta-model entities: indicating the correspondence between the meta-model used for the original model and how they were linked/transformed to the overall meta-model used in this research, as specified in chapter 5.

The following table describes in more detail the various attributes that have been captured for each of the models in the generic enterprise model database.

Model ID	**A short word or acronym used to refer to the model in the text. Also used as file name and for quick reference.**
Model Name	The official full model name, usually including to model owner.
Logo	A graphic logo of the model or, failing that, the model owner/creator.
Model Description	A description in about 200 words of the model, its background and other contextual information of interest.
Author / Copyright owner	The individual who (or institution which) created the model or owns the copyright.
Date (last change)	Date of last model update.
Primary source type	The type of source material which was used as the basis for the model capture (e.g. printed diagrams, source code in electronic format etc.)
Primary source reference	Printed source: bibliographic reference for the primary model source. On-line source: URL
Secondary sources	Alternative sources from which the model can be obtained.
Reference discipline	The reference discipline from which perspective the model was built (see chapter 4)
Modelling notation	The basic notation in which the source model is presented. This varies from natural language descriptions to more structured textual listings (e.g. in KIF) to various graphical notations such as data-flow diagrams and UML.
No. of concepts	The number of entities captured
No. of relationships (excl. generalisation)	The number of relationships between entities, excluding the generalisation relationships.
Directed graph?	Whether the relationships are directed or not i.e. whether the "from" and "to" entities are interchangeable. Where non-directed graphs can, in principle, be transformed into directed graphs by replacing each non-directional relationship with two directional relationships (one in each direction), this would double the number of relationships in a given model and increase its apparent complexity without basis in the initial model analysis effort.
No. of generalisation relationships	The number of generalisation relationships. Minimum is 0, where no super/sub type construct is used, but may theoretically exceed the number of concepts in the case of multiple inheritance.
Depth of inheritance tree	The maximum number of nodes that can be traversed in an inheritance tree in a given direction. A depth of 1 means a flat i.e. no tree.
Multiple inheritance?	Whether multiple inheritance (more that one super-type for any given concept) is used or not.
No. of grouper constructs	How many grouper constructs are used. Often given in the format of n : m : q where n = number of highest level grouper constructs; m = number of grouper constructs on the second-highest level etc. Typical grouper constructs are book chapters or individual diagrams. Grouper constructs encompass both relationships and entities. Entities (and, sometimes, relationships) can belong to more that one grouper

	construct.
No. of levels in grouping hierarchy	How many levels in the grouping hierarchy? 1 = no grouping; 2 = one set of grouping constructs into which the "basic level" entities are grouped; 3 = 2 levels of grouping constructs
Highest level groupings	A list of the names of the highest (and sometimes second-level) grouper constructs.
Meta-model mapping	Indicates the concepts in the meta-model of the captured model which were used to capture the model i.e. a translation from the constructs in which the captured model is presented to the meta-model used for generic enterprise model database.
Entity	The concept or construct that corresponds to an entity.
Name	The identifier for the name of an entity.
InternalCode	Any internal code or identifier used in the capturing model to reference model entities.
DefinitionOrDescription	How definitions or descriptions of entities are identified.
AttributeName	The concept used to identify the names of entity attributes or fields.
Relationship	The concept or construct that corresponds to a relationship.
Name	The identifier for the name of a relationship
InternalCode	Any internal code or identifier used in the capturing model to reference model relationships.
DefinitionOrDescription	How definitions or descriptions of entities are identified.
FromEntity	The construct used to indentify the entity from which a relationship departs (the "from node")
FromRoleName	The identifier for the name for the FromEntity.
FromCardinality	How the cardinality for the FromEntity is shown.
ToEntity	The construct used to indentify the entity to which a relationship departs (the "To node")
ToRoleName	The identifier for the name for the ToEntity.
ToCardinality	How the cardinality for the ToEntity is shown.
RelationshipType	Where used, which types of relationships are used in the model.
Generalisation	What construct is used to indicate a structural generalisation or sub/super-type relationship
Unmapped concepts	Important constructs or concepts that are present in the model which have no equivalent in my meta-model and have therefore not been captured in the database.

Blank spaces indicate null values. A null value is, as in most databases, an overloaded concept. In this case, this typically means that an attribute is not available or is not applicable. In a few cases, it indicates an attribute that is not relevant or could not be captured.

A.1 AIAI's Enterprise Ontology

Model ID	AIAI
Model Name	The Enterprise Ontology
Logo	
Model Description	The Enterprise Ontology is a collection of terms and definitions relevant to business enterprises. It was developed as part of the Enterprise Project, a collaborative effort to provide a framework (including a method and computer tools) for enterprise modelling. The idea was to provide one set of terms and definitions which adequately and accurately covers the relevant concepts in the enterprise modelling domain. Its main purpose was to facilitate communication between different parties, including users, developers and computer systems. The Enterprise Ontology was completed in 1996 in a natural language format and subsequently formalized, with minor modifications, in Ontolingua in 1998.
Author / Copyright owner	Artificial Intelligence Applications Institute (AIAI), University of Edinburgh (UK). The two lead authors are Mike Uschold and Martin King.
Date (last change)	1998
Primary source type	Electronic: KIF format
Primary source reference	http://ontolingua.stanford.edu KSL (Stanford University Knowledge Systems Laboratory) Ontolingua Server
Secondary sources	[USCH98] Mike Uschold, Martin King, Stuart Moralee and Yannis Zorgios. The Enterprise Ontology. The Knowledge Engineering Review, Vol. 13 (1998). Special Issue on Putting Ontologies to Use (eds. Mike Uschold and Austin Tate). This contains the final natural language version; KSL Ontolingua Server: CML, LOOM, GF (& KIF) format.
Reference discipline	Ontology
Modelling notation	KIF (text)
No. of concepts	94
No. of relationships (excl. generalisation)	73
Directed graph?	No
No. of generalisation relationships	98
Depth of inheritance tree	6
Multiple inheritance?	Yes (11 out of 98)
No. of grouper constructs	5 : 86 (concepts) - in natural language version only
No. of levels in grouping hierarchy	2
Highest level groupings	5: Activity; Strategy; Marketing; Organisation; Time.
Meta-model mapping	
Entity	Class {or} Object
Name	Class
InternalCode	
DefinitionOrDescription	Documentation
AttributeName	Template-Face-Value
Relationship	Relation {or} Function {or} <KIFcode>
Name	Relation
InternalCode	
DefinitionOrDescription	Documentation
FromEntity	Range

FromRoleName	
FromCardinality	
ToEntity	Domain
ToRoleName	
ToCardinality	Arity {or} Function-Arity
RelationshipType	
Generalisation	Subclass-Of
Unmapped concepts	<KIF axioms (constraints)> Template-Face-Value Value & Cardinality

Instead of using the natural language version provided by [USCH98], the formal version of the AIAI Ontology was downloaded from the KSL Ontolingua server. The download was available in the following KR languages: CML, LOOM, GF and KIF. The KIF version was used for model capture.

A typical code example of an AIAI concept is the following[1]:

```
(defrelation Employment-Contract
    (Template-Facet-Value Value-Type Employee Employment-Contract Person)
    (Template-Facet-Value Cardinality Employee Employment-Contract 1)
    (Template-Facet-Value Value-Type Employer Employment-Contract Legal-Entity)
    (Template-Facet-Value Cardinality Employer Employment-Contract 1)
    (Subclass-Of Employment-Contract Eo-Entity)
    (Class Employment-Contract)
    (Arity Employment-Contract 1)
    (Documentation Employment-Contract
    "an agreement between a Legal-Entity and a Person whereby the Legal-Entity employs the
Person"))
```

The KIF reserved word 'Subclass' indicates an IS-A relationship between Employment-Contract and Eo-Entity. There are two facets which really represent objects by reference: an Employee-Contract refers to ('Cardinality') 1 Employee of the type ('Value-Type') Person and 1 Employer of the type Legal-Entity. These result in an implied additional 'definitional' or 'attribute' (1:N) relationships between Employee-Contract, and Employee or Employer respectively. The entity's 'documentation' was also captured.

Slightly more complex relationships are found where the semantics of a concept is captured using KIF logic statements. This is an example:

```
(defrelation Purpose-Holder
    (?purpose-holder) :=
    (and (exists (?purpose) (Hold-Purpose ?purpose-holder ?purpose))
    (Actor ?purpose-holder)))
[...]
(Documentation Purpose-Holder
"The Actor in the Hold-Purpose Relationship")
```

The definition of the concept of Purpose-Holder is a specialisation of the semantics as found in the relationship captured from the 'defrelation' of Hold-Purpose: "a Relationship between an Actor and a State-Of-Affairs Affairs whereby the Actor wants, intends, or is responsible for the full or partial achievement of the State-Of-Affairs" involving the supertypes "State-Of-Affairs" and "Potential-Actor".

The definition of Strategic-Purpose includes a much more specific axiom and can be seen as a constraint. Note also the definition of the recursive relationship of Strategic-Purpose at the bottom.

```
(defrelation Strategic-Purpose
    (?strategic-purpose) :=>
    (exists (?actor) (Hold-Purpose ?actor ?strategic-purpose))
    :axiom
    (and (Subclass-Of Strategic-Purpose Purpose)
    (Class Strategic-Purpose) (Arity Strategic-Purpose 1)
    (Documentation Strategic-Purpose
    "A purpose held by an Actor that is declared to be of 'Strategic' importance. A Strategic-Purpose
may Help-Achieve another Purpose, and thus be lower-level with respect to that Purpose. However,
any such Purpose that is higher-level than at least one Strategic-Purpose must itself be a Strategic-
Purpose.")))
```

[1] KIF code fragments have been edited and formatted for demonstrational purposes.

```
(forall (?purpose ?sgc-purpose)
  (=>
    (and (Strategic-Purpose ?sgc-purpose)
         (Help-Achieve ?sgc-purpose ?purpose))
    (Strategic-Purpose ?purpose)))
```

A typical code example of a `proper' AIAI relationship is the following:

```
(defrelation Hold-Authority
      (Range Hold-Authority Activity-Spec)
      (Domain Hold-Authority Potential-Actor)
      (Relation Hold-Authority)
      (Arity Hold-Authority 2)
      (Binary-Relation Hold-Authority)
      (Documentation Hold-Authority
      "A Relationship between an Actor and an Activity-Spec whereby the
Actor has the right to perform the Activities as specified, i.e. to Execute
the Activity-Spec."))
```

This "Range" and "Domain" KIFwords provided the two entities between which the "Defrelation" relationship holds. The "Documentation" was also captured.

Remarkably, two cases of "subrelation" were also specified: (Subrelation-Of Have-Skill Have-Capability) and (Subrelation-Of Intended-Purpose Specified-Effect). Additionally a small number of 7 KIF `functions' (returning a state of For-Sale or Sale) and 10 KIF `instances' were defined. The former were interpreted as model relations and the latter were ignored since they related mainly to KIF axioms and constraints. Only one `inverse' relationship was defined (Manages -> Managed-By).

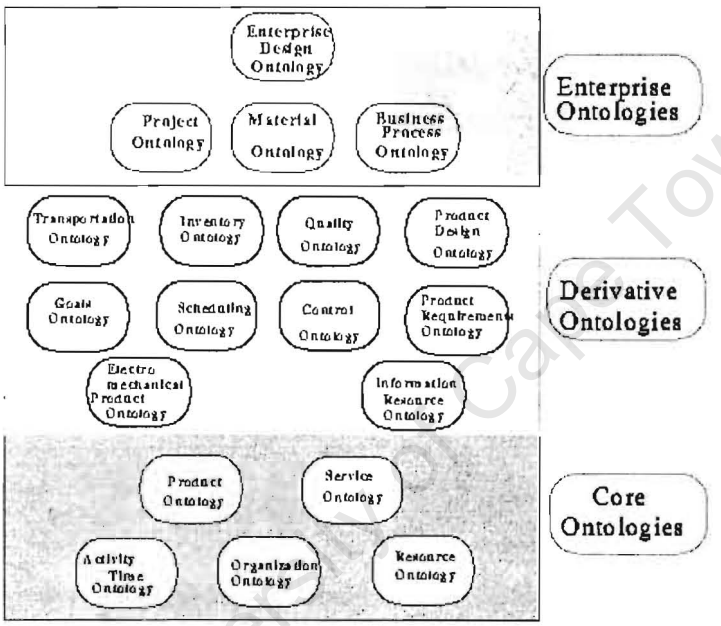
A small number of the entities and relationships were meta-constructs.

A.2 EIL's TOVE Ontologies

Model ID	TOVE
Model Name	The TOVE ("Toronto Virtual Enterprise") Ontologies
Logo	
Model Description	The goal of the TOVE (TOronto Virtual Enterprise) project is to create a data model that has the following characteristics: 1) provides a shared terminology for the enterprise that each agent can jointly understand and use, 2) defines the meaning of each term (aka semantics) in a precise and as unambiguous manner as possible, 3) implements the semantics in a set of axioms that will enable TOVE to automatically deduce the answer to many "common sense" questions about the enterprise, and 4) defines a symbology for depicting a term or the concept constructed thereof in a graphical context. The aim is to create a formal and computable terminology and semantics of activities and resources. The TOVE ontology is an ongoing research project; typically each ontology is a doctoral output. To date, not all sub-ontologies have been completed e.g. the electro-mechanical product, information resource, quality, service ontologies. The final version will comprise 19 separate ontologies. Some of the ontologies were subsequently included in the PSL specification.
Author / Copyright owner	Enterprise Integration Laboratory, University of Toronto, Canada. The two lead authors are Mark S. Fox and Michael Gruninger.
Date (last change)	1999
Primary source type	Electronic: KIF & Prolog format
Primary source reference	http://www.eil.utoronto.ca/tove/ Enterprise Integration Laboratory
Secondary sources	Many papers for each of the ontologies e.g. [FOX93; FADE94; FOX95; KIM95; LIN96].
Reference discipline	Ontology, Artificial Intelligence, Knowledge-Based Reasoning
Modelling notation	KIF, Prolog (text)
No. of concepts	564
No. of relationships (excl. generalisation)	1216
Directed graph?	No
No. of generalisation relationships	72
Depth of inheritance tree	4
Multiple inheritance?	Yes (5 out of 72)
No. of grouper constructs	11 (ontologies) : 74 (axiom sets) : 564 (concepts)
No. of levels in grouping hierarchy	4
Highest level groupings	(Level 1): 4 Enterprise + 10 Derivative + 5 Core Ontologies (Level 2): 11 ontologies: Activity/State/Time; Enterprise; Goals; Intended Action; Inventory; Material Flow; Operating Strategies; Organisation; Resource; Scheduling; Transport. (8 ontologies not yet completed/available: Project; Quality; Product Design; Product Requirements; Electro-mechanical Product; Information Resource; Product; Service)
Meta-model mapping	
Entity	Relation {or} Relationship {or} Class {or} Function
Name	Def-<x> {or} Define-<x>
InternalCode	
DefinitionOrDescription	<HTML text marked up between <i> and </i> tags >
AttributeName	Template-Slots
Relationship	<Derived from arguments for Def-x or KIF / Prolog code>

Name	
InternalCode	
DefinitionOrDescription	
FromEntity	Def-<x> {or} Define-<x>
FromRoleName	
FromCardinality	
ToEntity	<Any domain argument or other defined concept in definition or Prolog code>
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	Subclass-Of
Unmapped concepts	<KIF & Prolog definitions / axioms> Slot-Cardinality & Slot-Value-Type

A large number of articles have been written which document the various ontologies which make up TOVE ontology collection. The complete set of proposed TOVE ontologies is shown in the figure below:



Each existing ontology appears to have been developed by one or more doctoral students. The full documentation of these ontologies is therefore contained in their theses although a number of (mainly unpublished) papers summarize some of the ontologies e.g. [FADE94; FOX95; KIM95; LIN96].

The TOVE ontology is an ongoing research project and, to date, not all sub-ontologies have been completed e.g. the electro-mechanical product, information resource, quality, service ontologies. All source code (in KIF and/or prolog) for the available ontologies were downloaded from the TOVE website http://www.utoronto.ac.ca/EIL. After removal of duplicate and non-relevant files 103 source code files were used, totalling 264KB of code. These files were decoded partly manually partly automated in order to produce the required model information.

A search for all (Define-xxx) statements was made to isolate all concepts. An inconsistency was discovered in the used of the (define-) keyword: some ontologies would use "defrelation" and other ontologies "define-relationship" The following types of concepts were defined:

- Relations e.g. side_effect_product (from a resource to an action) but warehouse and repairable where also defined as relations.
- Classes e.g. production_sequence, inventory or agent_volume_schedule_deadline
- Functions e.g. move_batch or modify_deadline
- Theories typically containing axioms for specific ontologies.
- Some instances were defined: resource point "rp" and agg-demand "total_committed".

No strong semantic distinction could be discerned between classes and entities e.g. "warehouse" is defined as a single-argument relation, " nonrepairable" as a class, and "inventory_contains" as a function. The distinction seemed to have been driven more by technical implementation issues.

Subclasses, giving rise to the IS-A relationships, were easily identified by parsing the source code for the "(Subclass-of x y)" construct.

The grouping constructs were driven by the hierarchical file structure as contained on the TOVE website. There was quite a bit of inconsistency in the naming of the files.

Extracting the other relationships was the most difficult task, since this could not be automated. The following gives the sample code for two processor action resources.

Resources for Processor Actions

An object ?r is a processor resource for an activity ?a iff ?a is a processor action which uses ?r.

```
(defrelation processor_resource (?r ?a) :=
(and   (processor_action ?a)
       (uses ?a ?r))
```

An object ?r is an input material for an activity ?a iff ?a is a processor action which consumes or modifies ?r.

```
(defrelation input_material (?r ?a) :=
(and   (processor_action ?a)
       (or    (consumes ?a ?r)
              (modifies ?a ?r))))
```

Definitions as shown in italics above (usually a paraphrasing of the code) were captured for 111 entities.

The processor_resource entity (a relation) is linked by *definition* to two other entities: the processor_action (a class) and uses (a relation). The relation input_material is thus linked to processor_action, consumes and modifies. Trivial links to the entities holds or poss were not captured. 4696 lines of source code had to be parsed in this manner, much of which could not be automated. In addition to ontology *definitions*, a number of axioms were also included, which were similarly "mined" for relationships.

The code example below gives an example of multiple inheritance: tove_activity inherits from activity, resource_based and capacity_based. Also illustrated are three entity attributes (enabling, caused and status) which are captured as relationships from tove_activity to state and status_fluent respectively.

```
(define-class tove_activity
:own-slots
(Subclass-Of activity)
(Subclass-Of resource_based)
(Subclass-Of capacity-based)
:template-slots
((enabling          (Slot-Cardinality 1)
                    (Slot-Value-Type state))
 (caused (Slot-Cardinality 1)
                    (Slot-Value-Type state))
 (status            (Slot-Cardinality 1)
          (Slot-Value-Type status_fluent))
```

A.3 CYC Upper Ontology

Model ID	CYC
Model Name	The Cyc Ontology
Logo	
Model Description	The complete Cyc ontology is a multi-year project to attempt to formalize common sense. It has been under development since 1984 by AI pioneer Doug Lenat. The knowledge base is built upon a core of over 1,000,000 hand-entered assertions (or "rules") designed to capture a large portion of what is normally considered consensus knowledge about the world. It is tauted as the world's largest and most complete general knowledge base; CyCorp also has developed a common-sense reasoning engine to accompany with the knowledge base. In 1997, Cyc released the beta version of the Upper Cyc Ontology (version 2.1) i.e. the uppermost 2691 concepts (also called cyc constants, terms or units). A new, fully updated "open source" version, with approximately 6000 concepts, was scheduled to be released in the second half of 2001 but only appeared in Dec. 2002. The "generic enterprise model" has been distilled from the old version by selecting a sub-set of 777 Cyc constants which was deemed to be related to the enterprise domain. Although the selection process is subjective, including or deleting an extra few hundred constants would not really change the nature of the model, just its relative size.
Author / Copyright owner	CyCorp Upper ontology to be released as Open Source.
Date (last change)	1007
Primary source type	Electronic: CycL (CycLanguage)
Primary source reference	http://www.cyc.com/cyc-2-1/
Secondary sources	http://www.opencyc.org (to be released soon)
Reference discipline	Artificial Intelligence
Modelling notation	CycL (text) CycL is a formal language whose syntax derives from first-order predicate calculus (the language of formal logic). In order to express common sense knowledge, however, it goes far beyond first order logic. The vocabulary of CycL consists of terms. The set of terms can be divided into constants, non-atomic terms (NATs), variables, and a few other types of objects. Terms are combined into meaningful CycL expressions, which are used to make assertions in the CYC® knowledge base.
No. of concepts	777
No. of relationships (excl. generalisation)	1149 (421 from #$arg and 728 from #$comment)
Directed graph?	Yes
No. of generalisation relationships	873 (654 Genls & 219 Isa)
Depth of inheritance tree	10
Multiple inheritance?	Yes (182 out of 654 Genls)
No. of grouper constructs	43
No. of levels in grouping hierarchy	2
Highest level groupings	43: Fundamentals; Top Level; Time and Dates; Types of Predicates; Spatial Relations; Quantities; Mathematics; Contexts; Groups; "Doing"; Transformations; Changes Of State; Transfer Of Possession; Movement; Parts of Objects; Composition of Substances; Agents;

	Organizations; Actors; Roles; Professions; Emotion; Propositional Attitudes; Social; Biology; Chemistry; Physiology; General Medicine; Materials; Waves; Devices; Construction; Financial; Food; Clothing; Weather; Geography; Transportation; Information; Perception; Agreements; Linguistic Terms; Documentation. <only concepts from the underlined sections were selected>
Meta-model mapping	
Entity	CycCollection (Capitalized) {or} CycRelation (lowercase) {or} CycFunction (Fn appended)
Name	;;; #$...
InternalCode	
DefinitionOrDescription	#$comment "..."
AttributeName	
Relationship	#$arg... {or} #$resultIsa
Name	
InternalCode	
DefinitionOrDescription	
FromEntity	<1st argument #$...>
FromRoleName	
FromCardinality	
ToEntity	<2nd argument #$...>
ToRoleName	
ToCardinality	
RelationshipType	#$argIsa {or} #$argGenl {or} #$synonym
Generalisation	#$genls {or} #$genlPreds <Is-A-Type-Of> #$isa <Is-An-Instance-Of>
Unmapped concepts	#$genlInverse

The following shows two sample CycL code fragments to illustrate how the data was converted.

```
;;; #$CreditCard
(#$isa #$CreditCard #$MoneyTenderType)
(#$genls #$CreditCard #$TenderObject)
(#$genls #$CreditCard #$Card)
(#$genls #$CreditCard #$IDDocument)
(#$genls #$CreditCard #$OfficialDocument)
(#$genls #$CreditCard #$FinancialAccountTenderObject)
(#$comment #$CreditCard "A collection of plastic cards. Each element of #$CreditCard is a piece of plastic
that enables authorized users to spend the card-issuing company's money, drawn as a (usually unsecured) loan
through an associated instance of #$CreditCardAccount under a pre-arranged credit agreement. The credit card
company credits the vendor of the purchased goods or services and bills the card user, usually with interest.")
```

This example illustrates the following:

- ;;;#$: definition & naming of a CycCollection: "CreditCard" (the initial "C" capital indicates a collection)
- #$isa : A CreditCard is an instance (element) of the collection MoneyTenderType
- #$genls : A CreditCard is a (type of) TnederObject, Card, IDDocument, OfficialDocument and FinancialAccountTenderObject (illustrates multiple inheritance)
- #$comment : a definition / description of CreditCard. Note that the definition makes a reference to an additional concept namely CreditCardAccount, not indicated in the CycL definitional code but essential to capture.

```
;;; #$accountBalance
(#$isa #$accountBalance #$IntervalBasedQuantitySlot)
(#$arg1Isa #$accountBalance #$FinancialAccount)
(#$arg2Isa #$accountBalance #$Money)
(#$comment #$accountBalance "The predicate #$accountBalance is used to indicate the balance of a particular
account. (#$accountBalance ACCT BAL) means that the #$FinancialAccount ACCT has the balance BAL;
BAL is the amount of #$Money either owed by, or available to, the #$accountHolder (depending upon the type
of account).")
```

This example illustrates the following:

- ;;;#$account : definition & naming of a CycRelation: "accountBalance" (the initial non-capitalized "a" indicates a relation). Because many Cyc relations have more than two arguments, each CycRelation has been treated as a concept in its own right. This is also clear from the type of relationships that have been defined (many can have their own attributes, e.g. here it could be currency, date etc.)
- #isa IntervalBasedQuantitySlot will be ignored since IntervalBasedQuantitySlot is not part of the selected entities (i.e. entities seen as belonging to the enterprise domain).
- #$arg1Isa & #$arg2Isa reveal relationships from accountBalance to FinancialAccount and Money respectively.
- #$comment: a definition of the concept accountBalance, including a new relationship to accountHolder

The following will give a feel for the parsing and conversion process, as well as illustrate the cohesion within the selected concepts.

All of the 765 definitions provided by means of the #$comment (12 of the 777 entities did not have a definition) were parsed for references to other Cyc concepts as identified by the #$ tag. A total of 3742 CYC concepts were referred to. Of these 3742, 1410 referred to concepts outside the 777 entities; many of these were not even part of the (rest of the) Upper-Cyc Ontology i.e. not made publicly available at all. Of the remainder, many were duplicate references (i.e. the same concept repeated several times within the same definition) or self-references (re-stating the defined concept within the definition). After eliminating these, only 927 "genuine" relations remained. Of these, 199 were a duplication of the (421) relations already found from the CycL definition arguments (#$arg1 etc.), so a total of 728 "definitional relations" were added.

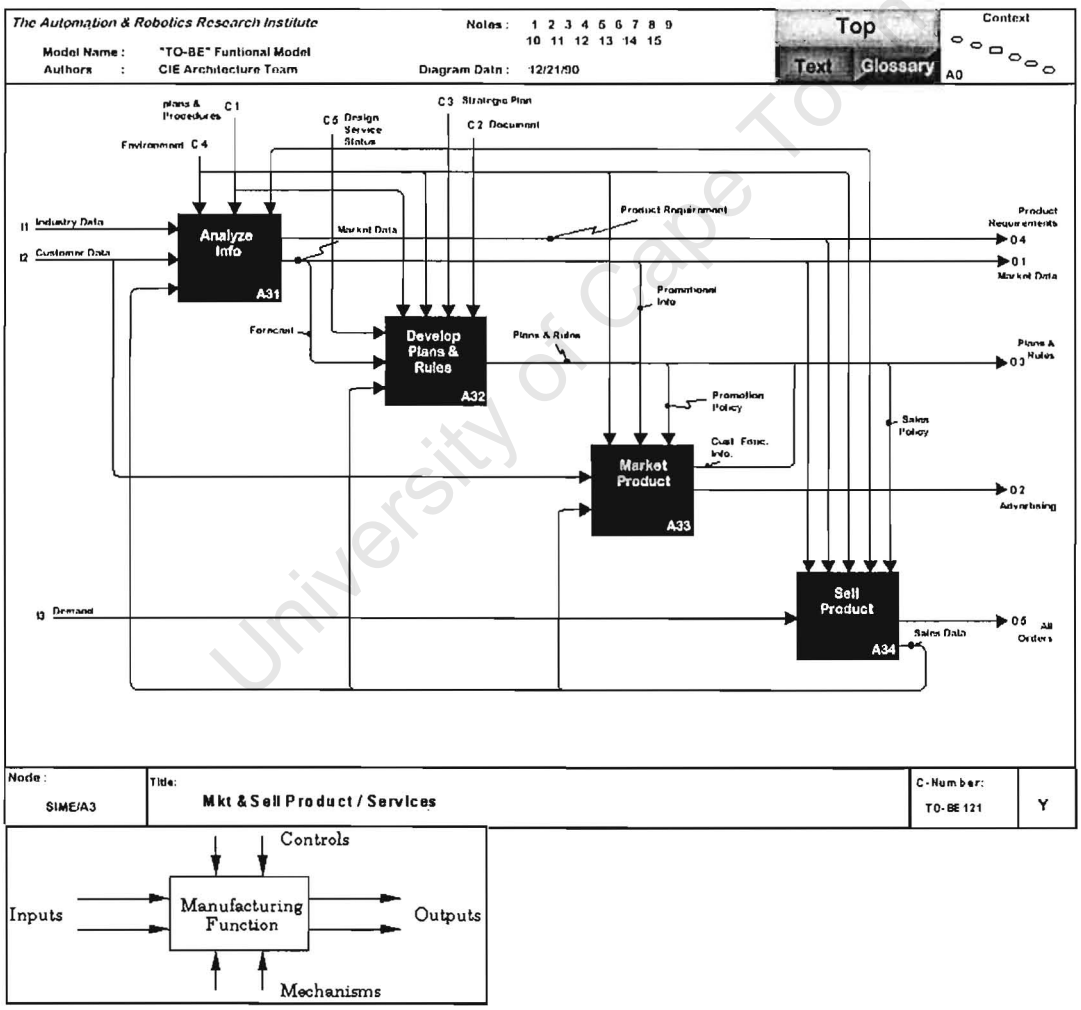
A.4 ARRI's Small Integrated Manufacturing Enterprise (SIME) Model

Model ID	ARRI
Model Name	ARRI's Small Integrated Manufacturing Enterprise Model
Logo	ARRI
Model Description	ARRI's Enterprise Engineering program's goal is to research and develop methods and tools to implement the integrated enterprise. It is developing enterprise reference models to provide a standard understanding of the manufacturing enterprise, using the IDEF0 methodology. Is has developed an enterprise model for the operations of a Small Integrated Manufacturing Enterprise (SIME, published 21 December 1990) and is busy developing an updated version of this model. These form part of a set of related models, including one for continuous enterprise improvement, enterprise transformation and enterprise performance management. The SIME operational activity is broken down into 6 subactivities,each of which is in turn broken down into approximately 4 subactivities.
Author / Copyright owner	Enterprise Engineering Program, Automation & Robotics Research Institute (ARRI), Arlington's College of Engineering, The University of Texas..
Date (last change)	1990
Primary source type	Printed: IDEF0 diagrams.
Primary source reference	http://arri.uta.edu/enteng/sime/a-0.htm http://arri.uta.edu/enteng/sime/sime.pdf
Secondary sources	
Reference discipline	Enterprise Engineering
Modelling notation	IDEF0 diagrams with associated text and glossary. Note: the Mechanism construct is not used in this model.
No. of concepts	128
No. of relationships (excl. generalisation)	197
Directed graph?	Yes
No. of generalisation relationships	70
Depth of inheritance tree	3
Multiple inheritance?	No
No. of grouper constructs	6
No. of levels in grouping hierarchy	2 (excluding the top-level "-0" context diagram)
Highest level groupings	6: Perform Strategic Planning; Manage Resources; Market & Sell Product/Services; Design Product/Services; Conduct Manufacturing Operations; Support Product.
Meta-model mapping	
Entity	Activity {or} DataFlow
Name	<Name Label>
InternalCode	<ActivityNumber for Activities>
DefinitionOrDescription	<Description>
AttributeName	
Relationship	<each ICOM connection>
Name	
InternalCode	
DefinitionOrDescription	
FromEntity	<Input/Control: Flow label; Output: Activity label>
FromRoleName	
FromCardinality	

ToEntity	<Input/Control: Activity label; Output: Flow label>
ToRoleName	
ToCardinality	
RelationshipType	Input {or} Output {or} Control {or} Aggregator
Generalisation	See capture notes
Unmapped concepts	

ARRI's SIME model is available as a set of IDEF0 diagrams with associated text and can be downloaded as a PDF file or as a hierarchical set of clickable GIF files embedded in HTML [http://arri.uta.edu/enteng/sime/nodetree1.htm]. A-0 is the context diagram containing the single "Operate A Small Integrated Manufacturing Enterprise", which explodes into A0, containing the 6 main activities (Perform Strategic Planning; Manage Resources; Market & Sell Product/Services; Design Product/Services; Conduct Manufacturing Operations; Support Product) and their ICOMs. Each of the main activities in exploded further into a detailed sub-diagram. The diagram below is an example, A3, detailing the activity "Market & Sell Product/Services".

In the IDEF0 notation, boxes represent activities and arrows data/information flows between the activities. Of the four possible types of IDEF0 ICOM data flows (see illustration below), the "Mechanism" flow is not used in the SIME model. All activities were captured as model entities. An critical decision was made to interpret each ICOM flow also as a model entity and *not* as a relationship. Initially, this may seem counter-intuitive, but this is



due to the subconscious association of "arrows" with "relationships" which is made by modelers, especially those familiar with the ERD and/or UML notations. The motivations for treating the IDEF0 flows as entities are as follows:

- Many flows represent substantive and important information documents e.g. in the diagram above: forecasts, market data, product requirements, plans & rules, strategic plan, customer data etc.
- Many of the flows (information packages) are combinations or aggregations of individual flows (smaller constituent information documents) e.g. in the diagram above the "Plans & Rules" consists of a promotional policy, customer education information and sales policy. These sub-flows merge or split off the main flow as required (see diagram). This behaviour is incompatible with a relationship.

- From a technical point of view, most flows may represent a "many-to-many" connection which would have to be resolved through a relationship-entity in normalization processes. In addition, one would be faced with n-ary relationships because many flows connect to a substantial number of activities.
- Most importantly: the semantic meaning of the flows corresponds to entities in other generic models.

As a consequence, the relationships were derived from connections between Activity-entities and Flows-entities, hence there were three types of (uni-directional) relationships: Inputs, Outputs and Control.

Two types of generalization relationships were inferred from the diagrams.

(1) Activity-entity generalizations.

Higher-level activities (e.g. A3: Mkt & Sell Product/Service) are considered to be super-types of their constituent activities (in this case A31, A32, A33 and A34). This is not entirely semantically correct since it is really a more comprehensive decompositional relationship: e.g. the high-level activity also includes the flows between A31...A34 and there is not necessarily full attribute inheritance. However, this is partially due to the semantic overloading of the IDEF0 meaning of "A3", since it represents both the A3 diagram (representation) and the A3 activity (semantics). Hence the translation requires a separation of the implicit semantic generalization relationship (A33 "Market Product" is a kind of A3 "Market & Sell Product/Service" activity) as well as a syntactical grouper relationship (A33 appears on the A3 diagram, which is an explosion/zoom from the A3 box on the A0 diagram).

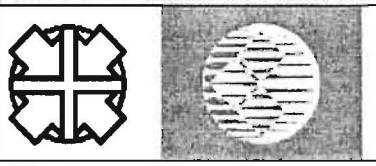
(2) Flow-entity generalizations.

Much of the same reasoning can be applied to the treatment of subflows and aggregated flows, although there is a slightly better case to be made to treat these as aggregation/composition relationships. However, the latter approach would complicate both the meta-model and the interpretation of the original model unnecessarily. This is especially so in the light of the fact that there is probably more attribute and method inheritance in the case of flow entities (characteristics and behaviours of "plans & rules" in general are likely to apply to (be inherited by) the "promotional policy" or "sales policy" in particular.

Because the generalization relationship in flow-entities is a structural relationship with specific semantic significance (inter alia, inheritance), it is not necessary to add a separate relationship to denote the connection between e.g. "plans & rules" and the splitting off of "promotional policy" (as a control flow for A33 market product): this can be inferred from the fact that "promotional policy" is a sub-entity of the "plans & rules" entity. A similar situation will be encountered in the dataflow diagrams of the Purdue model, but this time for the process entities, which would be the equivalent of activity-entities in the SIME model.

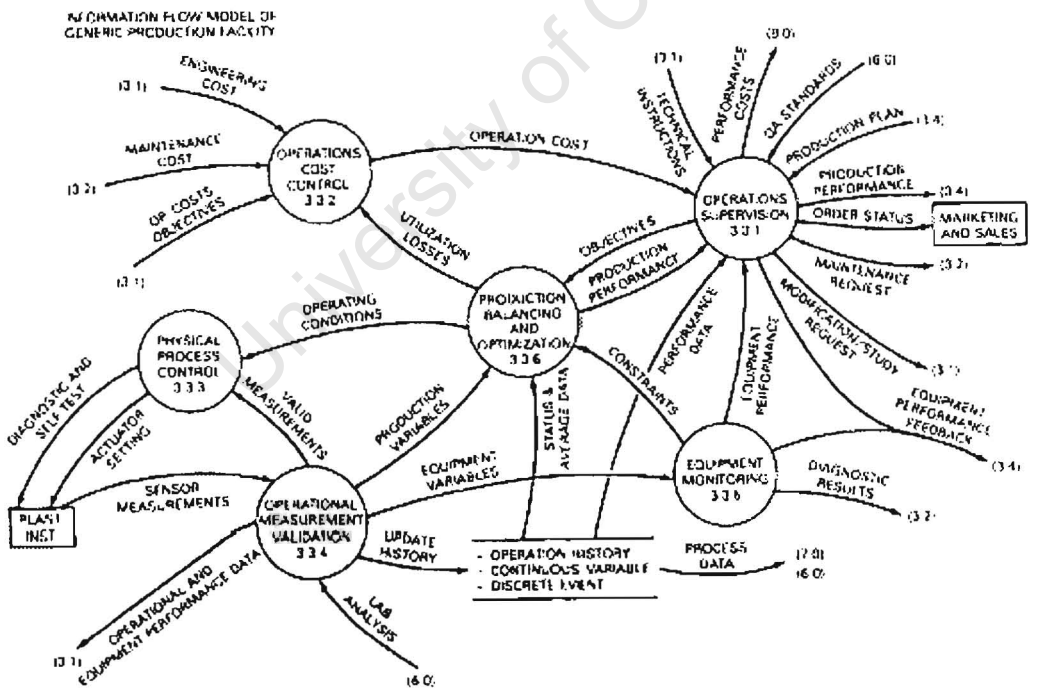
Generally, the SIME model showed great consistency between diagrams, although a few very minor spelling inconsistencies were encountered. The flows of "resource" & "spare" as ICOMs on the diagrams were not defined in the accompanying text, so the terms resources and spares were used instead. This may be incorrect (with the term definitions actually omitted), since both requirement & requirements were defined explicitly.

A.5 The Purdue Reference Model for Computer-Integrated Manufacturing

Model ID	Purdue
Model Name	The Purdue Reference Model for CIM
Logo	
Model Description	This is one of the earliest attempts to create a standard reference architecture within IS, and can be seen as one of the origins of the enterprise engineering discipline. The initial idea was to check which and to what extent each operation or function with a manufacturing organisation was automatable. The Workshop Committee created this model, using the now dated data flow and hierarchical modelling notations, in the hope of it becoming the foundation for a standard reference framework within the CIM community. The model was subsequently used in the standardization work of the Working Group 1 of subcommittee 5 of Technical Committe 184 (Industrial Automation Systems) of the ISO and is the background for the PERA (Purdue Enterprise Reference Architecture). The Purdue model consists of a set of data flow diagrams, annotated with a detailed lexicon of terms (for the "data dictionary"), and a scheduling and control hierarchy mapped to the diagrams. The Purdue Reference Model document also includes many additional inputs, such as the context of the model, the process of implementation, software requirements for CIM, the specification of interfaces and the role of the human in the automated CIM environments. Although these make up a significant portion of the publication, they are not relevant for the model capture.
Author / Copyright owner	Prepared by: the International Purdue Workshop on Industrial Computer Systems, Purdue Laboratory for Applied Industrial Control, Purdue University Adopted by: the Instrument Society of America. Lead author: Theodore J. Williams.
Date (last change)	1991
Primary source type	Printed: data flow diagrams and hierarchical models in textual form.
Primary source reference	http://www.pera.net/Pera/PurdueReferenceModel/ReferenceModel.pdf (scanned document) [WILL91]
Secondary sources	[WILL91] Williams, TJ (ed.). A Reference Model For Computer Integrated Manufacturing (CIM). A Description from the Viewpoint of Industrial Automation. CIM Reference Model Committee International Purdue Workshop on Industrial Computer Systems, The Instrument Society of America, Research Triangle Park, North Carolina, 1991 (2nd ed).
Reference discipline	Enterprise Engineering
Modelling notation	Printed: a blend of data flow and hierarchical models.
No. of concepts	106
No. of relationships (excl. generalisation)	224
Directed graph?	Yes
No. of generalisation relationships	103
Depth of inheritance tree	3
Multiple inheritance?	No
No. of grouper constructs	12 (see highest level groupings)
No. of levels in grouping hierarchy	3
Highest level groupings	9: Order processing; production scheduling; production control

	(further subdivided into: process support engineering; maintenance; operations control; operations planning); materials and energy control; procurement; quality assurance; product inventory; cost accounting; product shipping administration.
Meta-model mapping	
Entity	Process {or} Datastore
Name	Process Name {or} Datastore Contents
InternalCode	Process Code
DefinitionOrDescription	<Description>
AttributeName	
Relationship	Information Flow
Name	Flow Label
InternalCode	
DefinitionOrDescription	
FromEntity	Non-arrowed node
FromRoleName	
FromCardinality	
ToEntity	Arrowed node
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	See capture notes
Unmapped concepts	

Chapter 4 of [WILL91] contains all the data flow "graphs" which were used as the basis for model capture. The figure below illustrates the data-flow graph for process 3.3: Operations Control [WILL94, p.55]



The processes (circles), external entities and processes from other graphs (rectangles) and data stores (parallel lines) were all considered to be entities. The information (data) flows were treated as relationships. This is contrary to the practice adopted for ARRI's SIME model where the IDEF0 information flows were treated as separate entities. The following is the motivation for the chosen approach, to be read in parallel with the motivation for the SIME model:

- All data flows are directional from one to one process only (in SIME: n-ary with n often a large number, typically more than 10% of the total number of entities).

- The information flows often consists of "relatively small amounts of passed information", very much like the type of messages passed between objects or computer systems e.g. in the diagram above: process data, equipment variables, actuator settings, maintenance cost, sensor measurements. By contrast, in the SIME model, the information units almost always made up very substantial documents or sets of documents (e.g. policies) which would typically be embodied in separate physical documents.
- None of the information flows are composite flows i.e. split or combine along their route.

It must be noted that this is, to an extent, a subjective choice and all flows could indeed be modeled as entities. However, this is the case for any relationship in any model (refer to the discussion under the meta-model for this research) and adopting this stance for the Purdue model would merely have created a large number of entities with a corresponding set of twice as many (linking) relationships, without adding value to the model analysis.

With respect to the generalization and grouper constructs, the same approach as for the SIME model was adopted and a similar "overloading" of high-level processes as being both syntactic "zoomed-out" diagrams and decomposable into smaller sub-processes applies.

Unlike the SIME model, a fair number of inconsistencies was found, no doubt because the generation of the graphs seems to have been done manually (the SIME model was tool-supported). The following comments apply:

Two non-directional data flows (R107 & R219) were encountered. The direction of their flow was decided by semantic interpretation. Note that the graphs contained a couple of bi-directional arrows (mainly datastore queries), which were implemented as two uni-directionals.

Some new external entities crop up in the lower-level diagrams without appearing in the high-level diagram, e.g assigned work crews in diagram 3.2.

There are a number of unlabelled data flows.

In particular, there are a large number of problems/inconsistencies relating to flows spanning two diagrams:

- Differing names e.g. ANALYSIS DATA vs LAB ANALYSIS etc.
- Incorrectly labelled diagram codes e.g. 3.1 to 3.2.2 (on 3.3) is really from 8.1!
- The following "spanning" flows appear on one diagram but don't have counterparts on the other (corresponding) diagram

FROM	TO	NAME	DIAGRAM GROUP
R011	321	332	WORK REPORT
R012	321	332	DIAGNOSTIC AND SELF TEST REQUESTS
R017	311	325	INSTALLATION UPDATE
R025	67	332	PROCESS DEVIATIONS
R031	64	24	PRODUCTION VARIANCES

- Flows spanning 2 diagrams don't point out detailed subactivity, just overall diagram e.g. from 3.2.1 to 4.0 (instead of 4.1) or from 4.1 to 3.0 instead of 4.1 to 3.2.1

Full descriptions (definitions) for most processes were listed in separate tables and were scanned in using OCR.

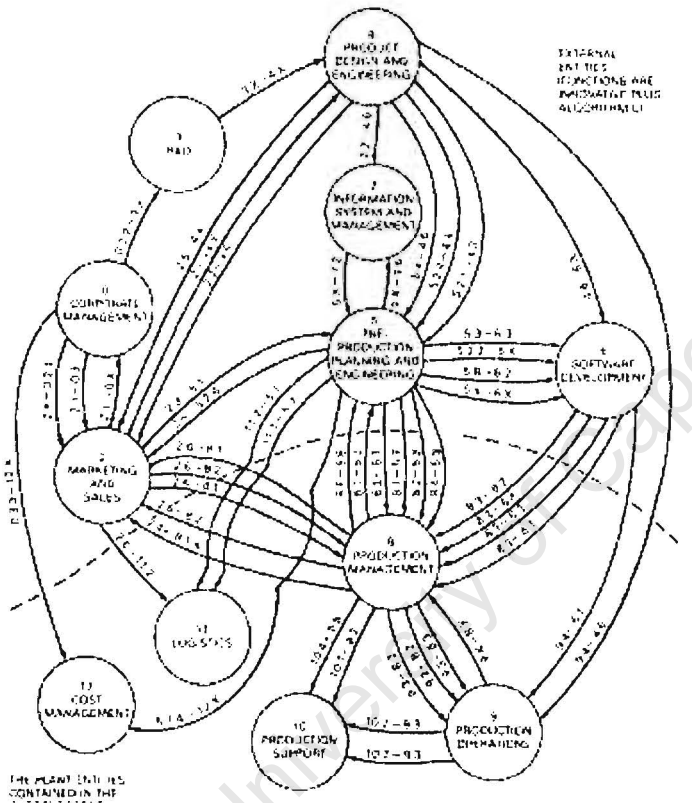
A.6 The Nippon Steel Corporation Industrial Automation System Model

Model ID	Nippon
Model Name	The Japanese Industrial Automation System Model
Logo	Nippon Steel Corporation
Model Description	The Purdue Reference Model document contains an appendix V which describes "an adaptation of a recent Japanese Industrial automation System Model (dated June 11, 1987, Anonymous)". Although the same notation is used as for the Purdue model, the underlying philosophy is very different and the resulting data-flow model, apart from being far less detailed, is "much different from that described in Chapter 4" (i.e. the Purdue reference model). Since the capture involved a small marginal effort, it is hereby included.
Author / Copyright owner	"Anonymous", Nippon Steel Corporation, Nagoya, Japan.
Date (last change)	1987
Primary source type	Printed: data flow diagrams and hierarchical models in textual form.
Primary source reference	Anonymous, Industrial Automation System Model, Nagoya Works, Nippon Steel Corporation, Nagoya, Japan (June 11, 1987). (Reference could not be traced.)
Secondary sources	http://www.pera.net/Pera/PurdueReferenceModel/ReferenceModel.pdf (scanned document) [WILL89]
Reference discipline	Enterprise Engineering
Modelling notation	Printed: a blend of data flow and hierarchical models.
No. of concepts	147 (many not partaking in relationships)
No. of relationships (excl. generalisation)	58
Directed graph?	Yes
No. of generalisation relationships	146
Depth of inheritance tree	5
Multiple inheritance?	No
No. of grouper constructs	4 : 12 : ...
No. of levels in grouping hierarchy	5
Highest level groupings	A. Corporate governance & management: corporate governance & management; corporate staff functions. B. Marketing and sales: marketing and sales. C. Research, development and engineering: R & D; product design and engineering ; preproduction planning & engineering; software development for production; information system and management. D. Production management, operations, quality assurance, and support, logistics, and cost management: production management; perform production operations; production support; logistics; cost management
Meta-model mapping	
Entity	Process {or} Datastore
Name	Process Name {or} Datastore Contents
InternalCode	Process Code
DefinitionOrDescription	<Description>
AttributeName	
Relationship	Information Flow
Name	Flow Label
InternalCode	
DefinitionOrDescription	
FromEntity	Non-arrowed node
FromRoleName	

FromCardinality	
ToEntity	Arrowed node
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	
Unmapped concepts	

The Purdue Reference Model [WILL91] contains appendix V with a "Japanese CIM model", originating from the Nagoya plant of the Nippon Steel Corporation. The original reference (very "non-specific" and anynomous) could not be found. The model, as presented in Appendix V, is relatively small, but could be added with relatively little effort and is a useful benchmark to compare against the Purdue model.

The meta-model and capturing procedure is identical to the one used for the Purdue Reference Model.



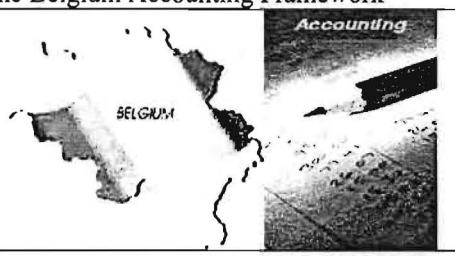
The processes were not specified in more detail in diagram format, but a hierarchical list was provided with the detail processes and their descriptions. This list yielded a relatively large number of entities (147) for the single diagram above, which contains only 47 relationships, hence many entities don't partake in specific relationships with other entities, except by "inheritance" of data flows from higher-level (parent) processes.

Some data flows (indicated with "SD" code in the hierarchical table) flow between more than 2 activities but, unlike in the ARRI model, they are modelled as (multiple) binary relationships, in accordance with the convention used in the Purdue model (and for the same reasons).

The following errors were found in to/from data flows:

- Process 8.1.4 doesn't exist - replaced with 8
- Process 7.6 doesn't exist - replaced with 7.5
- Headers in the table with processes sometimes repeat e.g. E018 & E019 = MARKETING AND SALES

A.7 The Belgium Accounting Framework

Model ID	BelgAcc
Model Name	The Belgium Accounting Framework
Logo	
Model Description	The accounting model is the oldest known general model that has been developed as the basis of a business information system. Not only are the oldest surviving writings, Mesopotamium clay tablets, assumed to be records of business transactions; but the "double entry" accounting model in its most basic form stems from Pacioli [1492]. National accounting bodies in most countries have adopted and/or expanded the basic model in specific ways, usually by means of specific government legislation or through the publication of standards and guidelines by the national accounting authority. The most detailed models have been developed in Western Europe, perhaps because of an arguably more pronounced culture of statutory public accountability. In 1972, the Commission of the European Community promulgated its 4th directive promulgated to provided a role model for harmonising the various national accounting standards. Belgium was the first country to translate the directive into specific legislation by means of a comprehensive set of laws on "de boekhouding en de jaarrekening van de ondernemingen" dating from 17-Jul-1975, 30-Mar-1976, 24-Mar-1978, 1-Jul-1983, 12-Jul-1989, 6-Aug-1993, 6-Apr-1995 as amended/supplemented by the Royal Decrees of 15-Dec-1978, 16-Jan-1986, 30-Dec-1991 and 27-Apr-1995. Typical transactions between these acccounts, exemplifying typical "relationships" in the accounting model, were captured from one of the leading accounting textbooks [REYN94]. These laws provide an extensive compulsary chart of accounts and set of financial statements to be used/completed by all mid-sized (who can use a subset) and large companies in Belgium. E.g. in 1992, 14,417 (large) companies had to submit a full set of accounts and 139,572 (medium-sized) companies an abbreviated set, to the Financial Statements Centre which is run under auspices of the National Bank of Belgium
Author / Copyright owner	Chart of Accounts: Belgian Law Sample Accounting Transactions: Reyns, Jorissen & Vanneste [REYN94]
Date (last change)	1994
Primary source type	Printed: chart of accounts & sample accounting transactions.
Primary source reference	Belgian Law: "Wet op de Boekhouding en de Jaarrekening van de Handels- en Industriële Ondernemingen", 17 July, 1975 as amended by subsequent laws and Royal Decrees. [REYN94] Reyns, Carl; Jorissen, Ann & Vanneste, Jacques. Inleiding tot Accountancy. UFSIA Universitaire Reeks Economie, Antwerp (BE): 1994.
Secondary sources	
Reference discipline	Accountancy
Modelling notation	Printed text: Chart of Accounts: in table form Sample Accounting Transactions: example texts and journal entries.
No. of concepts	470
No. of relationships (excl.	213

generalisation)	
Directed graph?	Yes
No. of generalisation relationships	462
Depth of inheritance tree	4
Multiple inheritance?	No
No. of grouper constructs	3 : 10
No. of levels in grouping hierarchy	4
Highest level groupings	Level 1: A. Balance Sheet; B. Income Statement; C. Control Accounts Level 2: 1. Owners equity & Provisions; 2: Establishment, Fixed Assets & Long Term Debt; 3: Stock (Inventory); 4: Short Term Creditors; 5: Cash & Financial Assets; 6: Expenses; 7: Revenue; (8 & 9: can be used for internal purposes); 0: Control Accounts (for off-balance sheet items)
Meta-model mapping	
Entity	Account
Name	Account Name
InternalCode	Account Number
DefinitionOrDescription	
AttributeName	
Relationship	Transaction
Name	
InternalCode	(Journal number)
DefinitionOrDescription	(Journal date, document reference)
FromEntity	Debit Account
FromRoleName	
FromCardinality	
ToEntity	Credit Account
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	As per Chart of Accounts
Unmapped concepts	

The source documents for the model derived from the Belgium Accounting Framework are the law on "de boekhouding en de jaarrekening van de ondernemingen" (the accounts and financial statements of companies) dating from 17-Jul-1975, 30-Mar-1976, 24-Mar-1978, 1-Jul-1983, 12-Jul-1989, 6-Aug-1993, 6-Apr-1995 as amended/supplemented by the Royal Decrees of 15-Dec-1978, 16-Jan-1986, 30-Dec-1991 and 27-Apr-1995.

These laws provide an extensive compulsary chart of accounts and set of financial statements to be used/completed by all mid-sized (who can use a subset) and large companies in Belgium. The following is a sample excerpt from the legally prescribed chart of accounts.

7 OPBRENGSTEN
70 Omzet
700 Verkopen
701 Dienstprestaties
708 Toegekende kortingen, ristorno's en rabatten (-)
71 Wijzigingen in de voorraden en in de bestellingen in uitvoering
712 in de voorraad goederen in bewerking
713 in de voorraad gereed product
715 in de vooraad onroerende goederen bestemd voor verkoop
717 in de bestellingen in uitvoering
7170 Aanschaffingswaarde
7171 Toegerekende winst

The chart of accounts provides the proscribed account names (e.g. "OPBRENGSTEN = income; "Omzet" = Turnover; "Verkopen" = Sales etc.) as well as account numbers (7, 70, 700, 701 etc.). The account numbers also provide an immediate clue to the hierarchical account structure i.e. accounts 7170 and 7171 are consolidated into account 717; accounts 712, 713, 715 and 717 are consolidated into 71 etc. Although, in accounting terms, this structure goes far beyond the "generalization" concept of modelling (refer to end-of-year posting and summary accounts), it subsumes the semantics of generalization in the sense that e.g. "sales" is a

type of "turnover" account, and "turnover" is a type of "revenue" account. Hence the generalization relationships were derived directly from the chart of accounts, using the equivalent of a ParentAccountCode=LEFT(ChildAccountCode,LENGTH(ChildAccountCode)-1) formula.

The names in the chart are often abbreviated, e.g. the full name of account 7170 is "Wijzigingen in de bestellingen in uitvoering - Aanschaffingswaarde" ("Changes in orders being processes – at cost price") with parts of the name "inherited" from 71 and 717 respectively.

Typical transactions between these accounts exemplify typical "relationships" in the accounting model. E.g. the sale of goods is recorded in a specific way and this represents a common relationship between the accounts (entities) concerned. Transactions were captured from one of the leading accounting textbooks in Belgium [REYN94]. Although it would have been possible to create an exhaustive list of possible relationships between accounts, it was decided that a textbook reference would be more authoritative and less subjective. The particular textbook was chosen, not only because it is one of the leading academic accountancy books, but also because it discusses each account category in a systematic way.

The following illustrates two sample transactions using the general ledger structure.

	Bank à Vastrentende effecten	5500	2812	2.500	2.500
	Bank Vastrentende effecten à Opbrengsten uit financiële vaste active	5500 2812	750	180 150	330

The first (empty) column is reserved for the (unique) journal ledger transaction number. The second column contains the name of the account(s) to be debited and credited; the latter is/are tabbed in and preceeded by an "à" prefix (abbreviation of "aan" = to). Columns 3 and 4 contain the respective account numbers and columns 5 & 6 the (sample) monetary amounts to be recorded against each account. Date and documentary reference for the transactions are generally omitted from the textbook, although they should appear in the real-world scenario. For the first transaction, a relationship from the entity "5500: Bank" to entity "2812: Vastrentende effecten" (fixed interest bearing stock) is recorded.

The second transaction represents a multi-legged accounting transaction. Although they affect more than two accounts, these are commonly recorded in one journal entry, in effect representing an n-ary relationship. The meta-model used in this thesis only allows for binary relationships so this transactions were split up in their components i.e.:

- Relationship 1: FromEntity (Debit) = 5500: Bank; ToEntity (Credit) = 750 Opbrengsten uit financiële vaste active
- Relatinoship 2: FromEntity (Debit) = 2812: Vastrentende effection; ToEntity (Credit) = 750 Opbrengsten uit financiële vaste active

A few transactions were four- or, in exceptional cases, even five-legged and an understanding of the underlying apportionment of monetary amounts was at times necessary to derive the correct binary relationships. It is noteworthy in this context that some simple computerized accounting systems (e.g. some spreadsheet-based systems or small-business systems such as "Finance Manager") also require binary journal entries i.e. they do not accept multi-legged transactions.

There appeared to be quite a few errors in the sample transactions, most of them relating to mistyped account numbers e.g. on page 269: account number 2190 instead of 2910. Another occasional error in the textbook was the use of monetary amounts in the columns reserved for the account numbers (e.g. 6000, 4110, 4400 on page 81). Finally, a problem was introduced because the legally prescribed chart of account allows the user to introduce further account subdivisions (detail) where deemed necessary or useful. Where this feature was used, the "parent" account was normally use e.g. on page 271, account number 4000 is used but the transaction was allocated instead to its parent 400. Sometimes, the legally proscribed chart of accounts gives an example of a detailed subdivision for one (the first) account in a given sub-category but not for others. Where this is the case, the semantics of the transaction preclude allocating the transaction to the parent account, so those transactions were ignored.

In all, 213 unique transactions against accounts present in the legal chart of accounts were found. A larger number of relationships obviously exists between the accounts, and quite a few relationships in the book were either against accounts not in the legal chart (i.e. more detailed accounts) or duplications/repeats of other transactions. Overall, these transactions used 123 of the 470 accounts in legal chart; although their parent accounts are obviously also affected, adding another 67 accounts to the active set. This leaves 278 accounts in the legal chart for which no examples exist in the textbook. Note that many smaller business only use a subset of the prescribed chart of accounts.

A.8 The U.S.B. Growth Model

Model ID	USB
Model Name	The U.S.B. Growth Model
Logo	
Model Description	The U.S.B. Growth Model is a menu-driven financial spreadsheet for use with Lotus 1-2-3 or clones. It will project the pro-forma financial statements, cash flow situation and performance measures of growth businesses for a five year period on a monthly basis. It has been designed to provide maximum flexibility although a conscious effort has been made to reduce the input requirements to an absolute minimum. The user can manipulate 27 to 72 variables and 10 parameters (not including the required historical data input). Sensitivity and target analysis capabilities are provided in addition to the extensive printed and graphical reports. All functional analysis and report generation can be selected from the user menus using a minimum number of keystrokes. It was developed as part of the requirements for the M.B.A. degree at the Graduate School of Business at the University of Stellenbosch. Although relatively small, it was pushing the limits of the personal computer hardware (RAM requirements) and software (Lotus 1-2-3) at the time of development (1988). Due to the generally prevalent 640K RAM constraint on PCs, version 2.1 was recompiled from a Lotus v.2 to Lotus v.1A format and released locally (in South Africa) as freeware. Although the model was meant to be a generic financial model for small and medium-sized businesses, its specific focus was to allow for 1 to 5 year forecasts of the maximum growth rate a business could sustain given certain financial decisions (profitability, financing mix, dividend policy, etc.). Despite its small size, it is considered to be a good exemple of generic financial models.
Author / Copyright owner	Author: Jean-Paul Van Belle Rights: University of Stellenbosch Version 2.1 released as freeware on 8-8-88.
Date (last change)	1988
Primary source type	Electronic: Lotus WKS and WK1 format
Primary source reference	Electronic file: "GROW.WKS"
Secondary sources	[VANB88] VAN BELLE J.P. The USB Growth Model: A Multi-variable Financial Computer Model for Growth Businesses. MBA Technical Report, University of Stellenbosch, 1988.
Reference discipline	Financial Management/Modelling Similarities with "System Dynamics" models
Modelling notation	Spreadsheet (mathematical formula) notation.
No. of concepts	147
No. of relationships (excl. generalisation)	239
Directed graph?	Yes
No. of generalisation relationships	258
Depth of inheritance tree	2
Multiple inheritance?	Yes
No. of grouper constructs	19
No. of levels in grouping hierarchy	4
Highest level groupings	Highest level: dependent & independent entities.

	Second level: INCOME STATEMENT; BALANCE SHEET; CAPITAL EMPLOYED; EMPLOYMENT OF CAPITAL; CASH BOOK; CASH IN; CASH OUT; STOCK BOOK ; INDICATORS; INDICES; FINANCIAL RATIOS (subdivided into: Liquidity; Activity; Leverage; Profitability; ROI Analysis); INPUT VARIABLES; PARAMETERS; HISTORY
Meta-model mapping	
Entity	Variable (independent or calculated by means of formula)
Name	Variable name
InternalCode	Cell Reference (for time-series: cell reference of first element)
DefinitionOrDescription	
AttributeName	
Relationship	Variable in formula calculation
Name	
InternalCode	Cell Reference
DefinitionOrDescription	
FromEntity	Independent variable
FromRoleName	
FromCardinality	Dependent variable
ToEntity	
ToRoleName	
ToCardinality	(Derived from mathematical operation in formula)
RelationshipType	(semantically derived)
Generalisation	
Unmapped concepts	Semantics of the equations / formulas

The U.S.B. Growth Model is a fairly simple financial spreadsheet model that was developed as part of the requirements for the M.B.A. degree at the Graduate School of Business at the University of Stellenbosch (Universiteit van Stellenbosch Bestuurskool) for Lotus 1-2-3. Despite its small size, it is considered to be a good and manageable exemple of generic financial models.

The following gives an example of a variables input screen.

	A	B	C	D	E	F	G	H
21	*** INPUT VARIABLES ***		Annual	|---------- Growth in:				----------|
22	**********	**Amount	growth	1988	1989	1990	1991	1992
23	Sales in Units/Mnth:	10000	20.00%	20.00%	20.00%	20.00%	20.00%	20.00%
24	Unit Sales Price:	$1.00	15.00%	15.00%	15.00%	15.00%	15.00%	15.00%
25	Unit Purchase Price:	$0.70	13.00%	13.00%	13.00%	13.00%	13.00%	13.00%
26	Unit Variable Cost:	$0.10	13.00%	13.00%	13.00%	13.00%	13.00%	13.00%
27	Fixed Cash Cost/Mth:	$1,000	11.00%	11.00%	11.00%	11.00%	11.00%	11.00%
28								
29	Payments in Month ->	0	1	2	3	4	5	Average
30	Accounts Receivable:	0%	0%	75%	25%	0%	0%	2.25
31	Accounts Payable:	0%	33%	33%	33%	0%	0%	2.00
32	Variable Costs:	0%	100%	0%	0%	0%	0%	1.00
33	Stockholding Period:	0%	25%	25%	25%	25%	0%	2.50
34	Seasonality Index =	Jan-Jun:	9091	9000	9100	9200	9600	9900
35		Jul-Dec:	10400	9400	11000	11200	10800	10600
36								
37	Intrest on Cash:	6.0%	|Dvd.Payout @		30%	|Tax Rate:		50%
38	Intrest on Overdraft:	16.0%	|Dvd.Frequency		2	Mths|GST Rate:		12%
39	==========	=========	========	========	========	========	========	=========

For example, row 25 contains the Unit Purchase Price. Although this looks like a single variable, there are actually two different variables: the "Initial Unit Purchase Price" (sub-type of both "CurrencyValue" and "IndependentVariable") and the "Annual Growth Rate of the Unit Purchase Price" (sub-type of "PercentageTimeSeries" and "IndependentVariable"). Other types of variables on this screen would be "Distribution Accounts Receivable" (Type: Distribution); "Interest Rate on Cash" (Type: PercentageValue); Dividend Frequency (Type: TimeIntervalValue); Initial Unit Sales (Type: UnitValue). These are all also independent variables.

	K	N
37	EMPLOYMENT OF CAPITAL	
38	Net Fixed Assets	=N39-N40
39	Fixed Assets	=M39
40	Depreciation	=M40+N127
41		
42	Current Assets	=SUM(N43:N45)
43	Stock	=-N110
44	Accounts Receivable	=M44+N120*$D110-N62
45	Cash	=IF(N85<0,0,N85)
46		
47	Current Liabilities	=SUM(N48:N54)
48	Accounts Payable	=M48+N123*$D109-N68
49	Creditors	=M49+N125-N69
50	GST Payable	=N120*$C110-N123*$C109+M50-N70
51	Dividends Payable	=M51+N24-N71
52	Income Tax Payable	=M52+N20-N72
53	Bank Overdraft	=IF(N85<0,-N85,0)

The relationships between variables were determined by means of the calculation formulas. An extract of some formulas is shown above.

A distinction was made between the following types of relationships, depending on the nature of the formula. Note that these relationship types are the same types as those typically encountered in dynamic systems modelling (refer to the section on Systems Theory).

Type	Description
Increases	A linear, positive influence, typically add, sometimes add value*constant + optional constant
Decreases	A linear, negative influence, typically deduct from
Complex influence	A composite influence, usually indicative really of a ternary relationship e.g =100*Var1/Var2
Conditional	Determines which calculation has to be followed e.g. FIFO vs LIFO, VAT vs GST etc. Note: this often subsumes an additional "*" relationship

The following relationships are some examples:

- Row 37: "EMPLOYMENT OF CAPITAL" is a heading of the financial statement (balance sheet) and represent a "grouper concept" for the entities "Net Fixed Assets", "Fixed Assets", "Depreciation", etc.
- Row 38: A "+" type relationship from "Fixed Assets" (a CurrencyTimeSeries-Type, N39 in the formula) and "Net Fixed Assets" (a CurrencyTimeSeries).
- Row 38: A "-" type relationship from "Depreciation" (CurrencyTimeSeries, represented by N40 in the formula) and "Net Fixed Assets.
- Row 39: A "+" type (recursive) relationship from "Fixed Assets" to "Fixed Assets".
- Row 42: Three "+" type relationships from "Stock", "Accounts Receivable" and "Cash" respectively, to "Current Assets".

The following are some additional notes in respect of the model translation.

Some duplicate names signifying different concepts were found in the model. Generally the context resolves the meaning of the concepts e.g. the label "Accounts Receivable" was used for two different concepts (entities): the ASSET (amount or closing balance owed by debtors, as appears on the balance sheet) or the PAYMENT (amount received during the month, as appears on the cash flow statement). A similar argument applies to e.g. "Accounts Payable" and "Dividends Payable".

Some re-engineering of the relations (as derived from the formulas) was not straightforward. The spreadsheet contained a sizable "work or intermediate calculation area". These intermediate calculations had to be resolved so that they did not show up as extra model elements. This sometimes meant tracing back calculations through several formulae to find the independent variables.

Some formulae are not straightforward e.g. to calculate the Dividend Payable, the following formula is used:

> DividendPayable = MaximumOf (DividendPayoutRatio*NetIncome AND "0")
> [Using the Lotus 1-2-3 @MAX formula]

This suggests a relation of the type "*complex influence*" from NetIncome to DividendPayable). However, the formula actually means

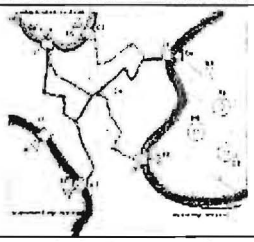
IF (NetIncome > "0") THEN DividendPayable=NetIncome*DividendPayoutRatio
ELSE DividendPayable = "0"

So the correct formula subtype is of the type "*choose or decision type*".

Two overall types of entities were found: independent and dependent. This separates input concepts (decision variables or historical values) from derived concepts (deterministically calculated).

Certain concepts occur twice or more e.g. the "Unit Sales Price" appears under Inputs as a value, as a growth rate and under indices. The motivation to separate these different meanings (and hence record them as separate entities) is the conceptual difference between the concepts i.e. value = actual (today's known value); growth rate = (gu)estimate; index = derived time series calculated using compound interest formula i.e. uncertain projected values the user wants to know and which is not immediately apparent from growth rate.

A.9 J.G. Miller's General Living Systems Model

Model ID	Miller
Model Name	J.G. Miller's General Living Systems Model
Logo	
Model Description	In 1978, James Grier Miller published his "Living Systems" book: a densely printed 1100 page tome. In it, he develops a general model of living systems, consisting of 19 subsystems (a number of years later, he added a 20th, the time) and states a large number of hypotheses relating to the system functioning. Particularly innovative was his attention to the information processing function, which accounts for half of the subsystems. He subsequently uses his systems model to check his hypotheses against a variety of living systems. His hierarchy of living systems stretches from the individual cell, via organs, organisms, individuals, societies etc. all the way up to the supranational system. Well over 100 pages are devoted to the organisation as a particular type of living system. A lot of the detail research is now somewhat dated, but his overall approach still rings very novel and original and it seems a pity that it had not been adopted more seriously in IS and other disciplines. However, after two decades, at least one IS-related research project is directly based on his model: [COFF97], although there would be significant scope for applying his theory to e.g. user-interfaces and communication theory. Although it is by far the smallest of the models surveyed here, it has been included specifically because of its originality and extreme generality.
Author / Copyright owner	James Grier Miller
Date (last change)	1978
Primary source type	Printed book
Primary source reference	[MILL78] Miller, James Grier. Living Systems. McGraw-Hill, New York, 1978.
Secondary sources	[MILL90; SKYT96]
Reference discipline	Systems Theory
Modelling notation	Natural language description. Very non-structured although a summary table of his 19 main subsystems can be found in [MILL78] p. 606-607 and a schematic diagram in [SKYT96] p.83.
No. of concepts	48
No. of relationships (excl. generalisation)	81
Directed graph?	Yes
No. of generalisation relationships	44
Depth of inheritance tree	5
Multiple inheritance?	No
No. of grouper constructs	
No. of levels in grouping hierarchy	
Highest level groupings	Highest Level: Matter-energy-information processors; Matter-energy processors; Information processors (& Environmen). Second Level: ; Reproducer; Boundary; Ingestor; Distributor; Convertor; Producer; Matter-energy storage; Extruder; Motor; Supporter; Input transducer; Internal transducer; Channel and net; Decoder; Associator; Memory; Decider; Encoder; Output transducer;

	Inclusion; Suprasystem
Meta-model mapping	
Entity	Subsystem
Name	<name>
InternalCode	<paragraph number>
DefinitionOrDescription	<description and examples>
AttributeName	
Relationship	Energy-Matter or Information Flow
Name	<textual description>
InternalCode	
DefinitionOrDescription	
FromEntity	<from text>
FromRoleName	
FromCardinality	
ToEntity	<from text>
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	<Subsystem of>
Unmapped concepts	<flow indicators/representative variables>

Miller's Living System model is the only example from the field of systems theory which was publicly available and had sufficient entities and relationships to be considered for this research. Two additional motivations for including the model were its originality (using a very different structure than most other business science theorists and practitioners) and the extreme generality of the model. The logo is taken from COFF97 and illustrates the interactions between the 9 *information* subsystems.

Although a small-diagrammatic version of the model is available in [SKYT96], the full model description is a rather lengthy descriptive exposition in "natural language" (common English) in [MILL78]. The following is a partial extract from the summary table. For the actual model capture, the text from pages 595 to 725 was used.

> 3.3 SUBSYSTEMS WHICH PROCESS INFORMATION
> 3.3.1 Input transducer.
> Such subsidiary organizations or groups as military intelligence agency or unit; guards, lookouts, fire watchers, meteorologists, astronomers, and others who observe and report upon environmental conditions or changes; market research department, persons that report on product or service acceptance or on economic and social trends which may affect the organization; sales department and others that take orders for the organization's products or services; intake department of social service or other organizations; medical personnel who take histories and examine patients on admission to clinics or hospitals; complaint department; legal department that obtains patents or licenses; radar, radio, and telephone operators; library acquisition staff; solicitors of money or credit for an organization; ticket sellers; dues collectors; tax collectors; and bank tellers; may be outwardly dispersed to consultants or researchers from another organization or downwardly dispersed to individual persons who transduce information inputs; artifacts include such communication and detection devices as telescope, field glasses, radar, radio, telephones, television
> 3.3.2 Internal transducer.
> Such subsidiary organizations or groups as make reports within an organization or ascertain needs, attitudes, or efficiency of components or subcomponents; spokesmen for components, like committee chail1l1en, department heads, union stewards and other officials, public opinion pollers, [...]

From the text above, the entity "Information-processing Subsystems" (internal code "3.3") can be seen to be a super-type (generalization relationship) of the entities "Input transducer" (code "3.3.1") and "Internal transducer" (code "3.3.2"). The entire paragraph [Such subsidiary organizations .... Telephones, television] serves as the "definition for the concept.

Since the "Input transducer" function can be "outwardly dispersed to consultants" etc. there is a direct relationship from "Input transducer" to "Environment". In addition, from the text it can be "inferred" that the information to be transduced comes from the environment, and therefore must cross the organization boundary (i.e. a relationship from "Boundary" to "Input transducer"). In fact, the more detailed text distinguishes between a Matter/Energy and an Information Boundary, so the latter is used as the "from Entity". The transduced information must be passed on to the rest of the organization and it is inferred that this happens via the (information) "Channel or net" implying a relationship between "Input transducer" and "Channel or net". The latter two relationships also occur explicitly in the [SKYT96] diagram.

The main relationships in Millers model are accounted for by the following general categories:

- Between Boundary and Environment;
- Between Distributor and each of the M/E (Matter/Energy) subsystems (for the transport of raw, intermediate and final products/energy requirements;
- Between Motor and the M/E subsystems (for mechanical power assistance);
- Between producer - M/E subsystems (for repairs);
- Between Channel/net and each of the Information subsystems (for the transport/distribution of the information);
- Between Decider and the M/E subsystems (monitoring & control function);
- Between many Subsystems and the Suprasystem or Environment (where functions are upward dispersed)

Some notes re the capturing process:

The model excludes marginal model updates from later publications (mainly consisting of the addition of "timer" subsystem and a diagrammatic representation illustrating the main flows between the entities).

The model as described by Miller includes input from Woodward, a systems theory researcher in organizational theory, specifically in the sections discussing the sub-classifications of production systems and the types of organising structures

Overall, the model required quite a bit of "subjective" interpretation e.g. various indicators for the processes of the matter-energy processing subsystems were interpreted as a relationship between "decider" (should really be the information channel/net) and each of those subsystems.

A.10 Bill Inmon's High and Mid-level Data Models

Model ID	Inmon
Model Name	Bill Inmon's High and Mid-level Data Models
Logo	BILLINMON.COM
Model Description	On his website, Bill Inmon "the prophet of Data Warehousing", provides two types of generic data models: industry-specific generic data models and functional generic data models. The industry specific generic data model reflects the basic business activities of a company engaging in commerce. The industry specific generic data models provided are: airline, banking, insurance, oil/gas, railroad and university. The functional generic data models reflect common functions done in any company. The functional generic data models provided are: the accounting, marketing, sales, corporate tax and contracts generic data model. The "industry-specific" manufacturing data model was also included as a generic data model, in line with the ERP models. The generic ones are used for this research. An organization that wishes to produce a complete model of the entire corporate environment could select several functional data models and combine them with an industry specific data model. The result would be a comprehensive data model of the entire corporation. His generic data models ("The business similarities between companies in the same industry are much more striking than the differences. As such, the data models that represent those businesses are likewise very similar.") consist of high- and mid-level generic data models. The mid-level diagrams contain important or typical attributes. The models appear to be based on his wide experience with implementing data warehouses across many industries. Although he is a co-author of the (previously published) Data Model Resource Book [SILV97], there seems to be little overlap between the models.
Author / Copyright owner	Bill Inmon
Date (last change)	1999-2000
Primary source type	Diagrams, available on-line
Primary source reference	http://www.billinmon.com
Secondary sources	
Reference discipline	Data Warehousing
Modelling notation	Own notation. There are two symbols used in the high level data model - an oval and an arrow. The oval surrounds the name of a major subject area, such as CUSTOMER or TRANSACTION. The arrow indicates a relationship between two major subjects. The arrowhead indicates cardinality. A single arrow indicates a 1:n relationship. A double headed arrow indicates an m:n relationship.
No. of concepts	427
No. of relationships (excl. generalisation)	241 (73 1:n arrows, 12 m:n arrows, 155 links)
Directed graph?	Yes
No. of generalisation relationships	220
Depth of inheritance tree	3
Multiple inheritance?	No
No. of grouper constructs	7 : 66
No. of levels in grouping hierarchy	3
Highest level groupings	First level: Accounting; Contract; Corporate Tax; Manufacturing; Marketing; Sales Second level: Account; Activity; Advertisement; Analysis; Assembly

	Line; Asset; Boycott; Business/Taxpayer; Calendar; Carrier; Cash Journal; Chart of Accounts; Contract; Contracts Administrator; Corporation; Customer; Debt; Deductions; Delivery; Depreciation; Employee; Employee Pay; Equipment; Equity; Expenses; Forecast; General Journal; General Ledger; Income; IRS; Job; Liability; Movement; Order; Package; Paid Preparer; Part; Payments; Penalty; Plant; Product; Product Catalog; Promotions; Refund; Report; Retailer; Sale; Sales Person; Schedule; Shareholder; Shipment; Stock; Stock Holders Equity; Stock Keeping Unit (SKU); Supplier; Supplier/Order; Tax; Tax Credit; Tax Form; Tax Return; Tax Schedule; Tax Shelter; Territories; Transaction; Vendor; Warehouse.
Meta-model mapping	
Entity	Subject {or} Data
Name	Label
InternalCode	
DefinitionOrDescription	
AttributeName	Field Label
Relationship	Relationship
Name	
InternalCode	
DefinitionOrDescription	
FromEntity	From arrow
FromRoleName	
FromCardinality	<by arrow type - see notes>
ToEntity	To arrow (point)
ToRoleName	
ToCardinality	<by arrow type - see notes>
RelationshipType	<1:n links {or} 1:n arrows {or} m:n arrows>
Generalisation	
Unmapped concepts	

All diagrams are available on-line from http://www.billinmon.com//library/models/models_home.asp after a simple registration procedure. By clicking any "subject" on the 7 high-level diagrams (containing the main relationship links), one can drill down to the mid-level data models which give more detailed entity level diagrams, including sub-entities, entity groupings and attribute listings.

The next page shows a screenshot of both a high-level diagram ("Marketing) and a mid-level diagram ("Territories").

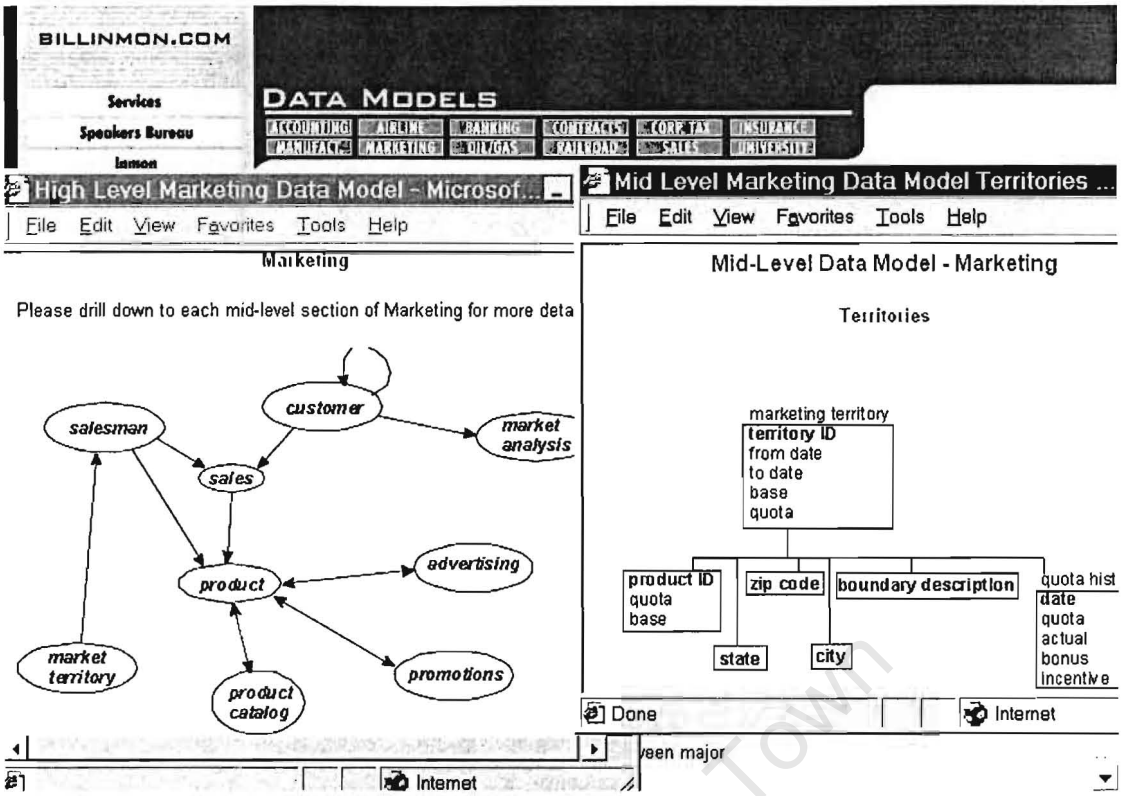
Although the notation in the high-level data model looks similar to a dataflow diagram, the arrows actually indicate types of relationships. The high-level diagram does not show derived relationships: although there is a relationship from a "market territory" to a "salesman" and from "salesman" to "product", the derived linkage from "market territory" to "product" is not shown, in proper modelling style. Note the bi-directional relationships between, for instance, product and promotions.

Relationships depicted with a single arrow-head are 1:n. Double arrow heads signify an m:n relationship. Where it made no business sense to break down an m:n relationship, Inmon did *not* normalize the relationship into two 1:n relationships and an intersection entity.

The conventions for mid level data modeling begin with the primary grouping of data. Each subject area found in the high level model revolves around one primary grouping of data. If the subject area CUSTOMER is recognized in the high level model, then there will be a corresponding primary grouping of data for customer that corresponds to the high level subject area.

The data models are not very well validated. The following are but some of the problems and inconsistencies which were encountered during the capture of the models.

- Inconsistent use of delimiters e.g. "date (actual)" versus "discounts – quantity" versus "supplier/order"
- There are lots of ad-hoc attributes, for example "production equipment – dimensions" include length and width, why not height?
- The way the address details are specified differ between the entities e-mail, address, phone, contact phone etc.



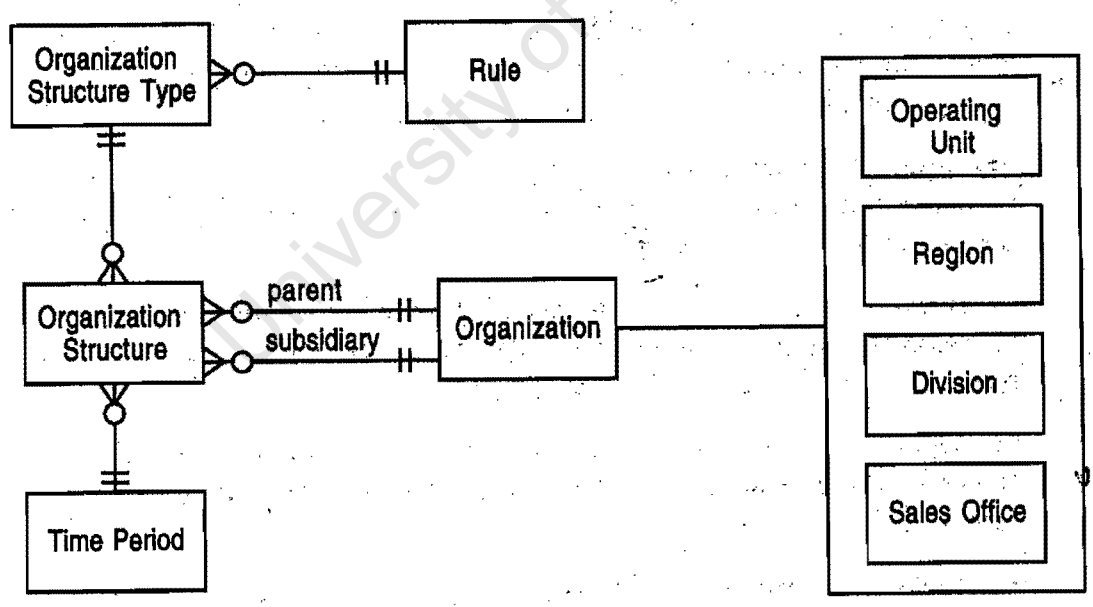
- There is an inconsistent use of options e.g. "receipt / no receipt" versus "follow up (yes/no)" versus "shared?" versus "current & long term" as two different attributes versus "long & short term".
- The model artificially inflates the number of attributes e.g. by using both "order number" and "order ID".
- There are many spelling mistakes e.g. "quaterly" ; "small buiseness"
- There appears to be a large coverage of the domain but it is generally of low quality
- There are numerous conflicts between the seven high-level models e.g.
  customer in sales: credit rating = commerical customer attribute
  customer in marketing: credit rating = (generic superclass) customer attribute
- The "Snapshot date" attribute in commercial & individual customer should move up to their common parent class (generic/abstract) "customer"
- Bill uses a number of non-regular business terms e.g. FILO instead of LIFO, warranty (where the meaning indicates "warrant") etc.
- Occasionally, there are some repeated fields (attributes) e.g. "in change in accounting year" (Business/Taxpayer) "change date" appears twice!
- The naming of data elements is very inconsistent e.g. (in Business/Taxpayer)
  change in accounting year: attribute = change date
  change in accounting method: attribute = date of change
  employee benefits: attributes = stop date ; qualification
  pension: attributes = end date ; qualifications

A.11 Fowler's Object-Oriented Analysis Patterns

Model ID	Fowler
Model Name	Fowler's Object-oriented Analysis Patterns
Logo	
Model Description	The book contains a large number of object-oriented analysis patterns, based on Fowler's consulting experience where he saw many problems (and their solutions) repeat innumerable times. The book is probably the best known book of OO analysis patterns. Although the book was published in 1997, much of the conceptual work was done earlier and the models are therefore presented in Oddell's notation (an EERD type notation) instead of UML diagrams. The majority of the patterns are very high level, although a number of chapters, and all chapters in the second half of the book, contain technical implementation or design-related patterns.
Author / Copyright owner	Martin Fowler
Date (last change)	1997
Primary source type	Printed diagrams in book.
Primary source reference	[FOWL97] Fowler, Martin. Analysis Patterns. Addison-Wesley: Reading (MA), 1997.
Secondary sources	
Reference discipline	Object-oriented Patterns
Modelling notation	Odell's Type Diagrams.
No. of concepts	121
No. of relationships (excl. generalisation)	131
Directed graph?	No (all patterns in the book are bidirectional associations)
No. of generalisation relationships	69
Depth of inheritance tree	4
Multiple inheritance?	No
No. of grouper constructs	6 : 49
No. of levels in grouping hierarchy	3
Highest level groupings	Highest Level: 2. Accountability; 3. Observations and Measurements; 4. Observations for Corporate Finance; 6. Inventory and Accounting; 8. Planning; 9. Trading. (Some other chapters are not relevant or generic enough.) Second Level: 2.10 Party Type Generalisations; 2.13 Accountability Types; 2.14 Operating Scope; 2.15 Post; 2.2 Party; 2.7 Organisation Hierarchies; 2.9 Accountability; 3.10 Observation & Measurement; 3.11 Dual time record; 3.12 Rejected observations; 3.13 Active Observation, hypothesis & projection; 3.14 Linking observations; 3.3 Conversion Ratios; 3.5 Compound Units Using Bags; 4.1 Objects of care; 4.10 Types of calculation; 4.11 Status types; 4.12 Comparative status types; 4.17 Dimension combination for calculated measurement protocols; 4.22 Adding a range to a phenomenon; 4.24 Range function; 4.5 Defining Enterprise segments using dimensions; 4.7 Measurements and measurement protocols; 4.8 Methods for calculated measurement protocols; 4.9 Kinds of observations; 6.21 Using account finding method; 6.23 Eligibility condition; 6.26 Accounting practice type; 6.27 Sources for a transaction; 6.28 Types of accounts; 6.29 Corresponding accounts; 6.3 Multi-legged account transactions; 6.30 Supporting inventories with the account model; 6.32 Multiple summary accounts; 6.4 Two legged transaction without entries; 6.5 Summary and detail accounts; 6.8 Posting rule with

	methods; 8.1 Properties of actions; 8.10 Relationship among action, plan and protocol; 8.11 Plan as a directed acyclic graph; 8.13 Action's use of resources; 8.15 Resource allocation for assets & consumables; 8.16 Links between observation, plan and action; 8.17 Start & outcome functions; 8.4 Types of actions; 8.6 Plan consisting of references to proposed actions; 8.7 Replacement plans; 8.8 Plans as actions; 9.3 Counter part & primary party for contract.
Meta-model mapping	
Entity	Type (See author's note in [FOWL97:314])
Name	Label
InternalCode	
DefinitionOrDescription	
AttributeName	
Relationship	Association
Name	Label (only added by Fowler when possible confusion arises)
InternalCode	
DefinitionOrDescription	
FromEntity	Relationship End
FromRoleName	Role
FromCardinality	Relationship End Symbol
ToEntity	Relationship End
ToRoleName	Role
ToCardinality	Relationship End Symbol
RelationshipType	Derived {or} <unspecified>
Generalisation	Subtype
Unmapped concepts	Constraints Incomplete partition (generalisation)

The following is a typical diagram from the book: Figure 2.7 from [FOWL97:23].



This was interpreted (partially) as follows. There is an unnamed relationship from the entity "Organization Structure" (From-Cardinality = [0..*]) to the entity "Organization Structure Type" (To-Cardinality = [1..1]). The relationship is bi-directional as are all (non-generalization) relationships in the book, so the TO and FROM entities could be swapped.

There are two relationships between "Organization Structure" and "Organization": one indicating the parent and the other the subsidiary organization. To avoid confusion, Fowler names all relationships with identical to & from entities.

The diagram shows a generalization relationship as a complete partition: "Organization" is the super-type of "Operating Unit", "Region", "Division" and "Sales Office".

Often alternative patterns are shown e.g. different ways of modelling an address book. Fowler often provides guidelines on when to use which alternative. For this research, the most complex diagram was always chosen.

The book contained a very few contradictions e.g.
fig 3.10: Observation 0..* <-> 0..1 Protocol
fig 4.7: Observation 0..* <-> 1..1 Protocol

Or
fig 8.14 Temporal Resource IS-A Specific Resource Allocation
fig 8.15 Temporal Resource IS-A Resource Allocation (short piece of line omitted)

Some relations have role names e.g. one in fig 6.27. Because they are so few, they were not recorded as role names but as a single slash-separated relationship name e.g. "consequences / sources".

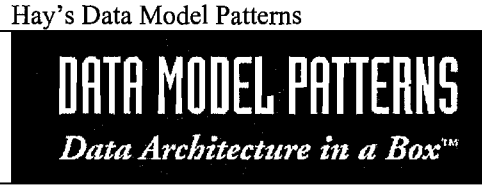
Patterns from the following chapters were not captured:

- Chapter 5: these are technical or implementation specific: object naming & object equivalence.
- Chapter 7: contains a worked example.
- Chapter 9: focusss on forex trading, only the generic diagram 9.3 concerning contracts was captured.
- Chapters 10 & 11: deal with (financial) derivatives and trading, so their domain is also not generic.
- Chapters 12 to 15: contain support patterns concerned with implementation or technical issues.

The captured models contain only one abstract type/entity namely "Operating Scope"

The models contain only three "derived" associations/relationships.

A.12 Hay's Data Model Patterns

Model ID	Hay
Model Name	Hay's Data Model Patterns
Logo	
Model Description	Hay's book "Data Model Patterns" describes a set of standard data models that can be applied to standard business situations. Although his models are described as patterns, and despite the fact that many of his models are indeed at a more abstract level than most data models, the models are not on the same high level as Fowler's. The book contains a large number of very well laid-out, clear and detailed diagrams with full documentation on the reasoning behind the models. There are well over 150 diagrams and tables. The model diagrams show relationship cardinalities and relationship role names. The book and its models are of such quality that they can easily serve as a showcase or benchmark for generic enterprise data modelling. Hay has followed up the book with some more advanced (higher level) patterns in [HAY98]. These were not included because they overlap partially with some of the models in [HAY96].
Author / Copyright owner	David C. Hay
Date (last change)	1996
Primary source type	Printed diagrams in book.
Primary source reference	[HAY96] Hay, David C. Data Model Patterns. Dorset House, New York, 1996.
Secondary sources	In electronic format on CD-ROM as an Oracle (template) database: Data Model Patterns: Data Architecture in a Box™ Available from http://www.essentialstrategies.com/patternscd/ See [HAY98] for some higher-level patterns.
Reference discipline	Data modelling
Modelling notation	EERD notation: CASE*Method by R. Barker, I Palmer & H Ellis. Entities = rounded boxes containing name and attributes; Sub-types = boxes contained in boxes; Relationship = connector line. Dotted = optional / Continuous = Mandatory; Conventional crowsfeet symbols for cardinalities; Role names provided as labels
No. of concepts	291
No. of relationships (excl. generalisation)	721
Directed graph?	No
No. of generalisation relationships	180 (19 multiple inheritance)
Depth of inheritance tree	5
Multiple inheritance?	Yes
No. of grouper constructs	8 : 83
No. of levels in grouping hierarchy	3
Highest level groupings	Highest level: 3 The Enterprise and Its World; 4 Things of the Enterprise; 5 Procedures and Activities; 6 Contracts; 7 Accounting; 9 Material Requirements Planning; 10 Process Manufacturing; 11 Documents Second level: 3.1 Parties; 3.7 Employment; 3.8 Organizations; 3.11 Geographic Location; 3.13 Geographic Structure Elements; 3.14 Organisational Structure; 3.16 Top-Heavy Hierarchy; 3.18 Reporting Relationships Between Parties; 4.2 Product Categories; 4.6 Manufacturing; 4.13 The Complete Model; 4.16 Asset Classes and Attributes; 5.3 A More Compact Way to Divide Activities; 5.4 Work

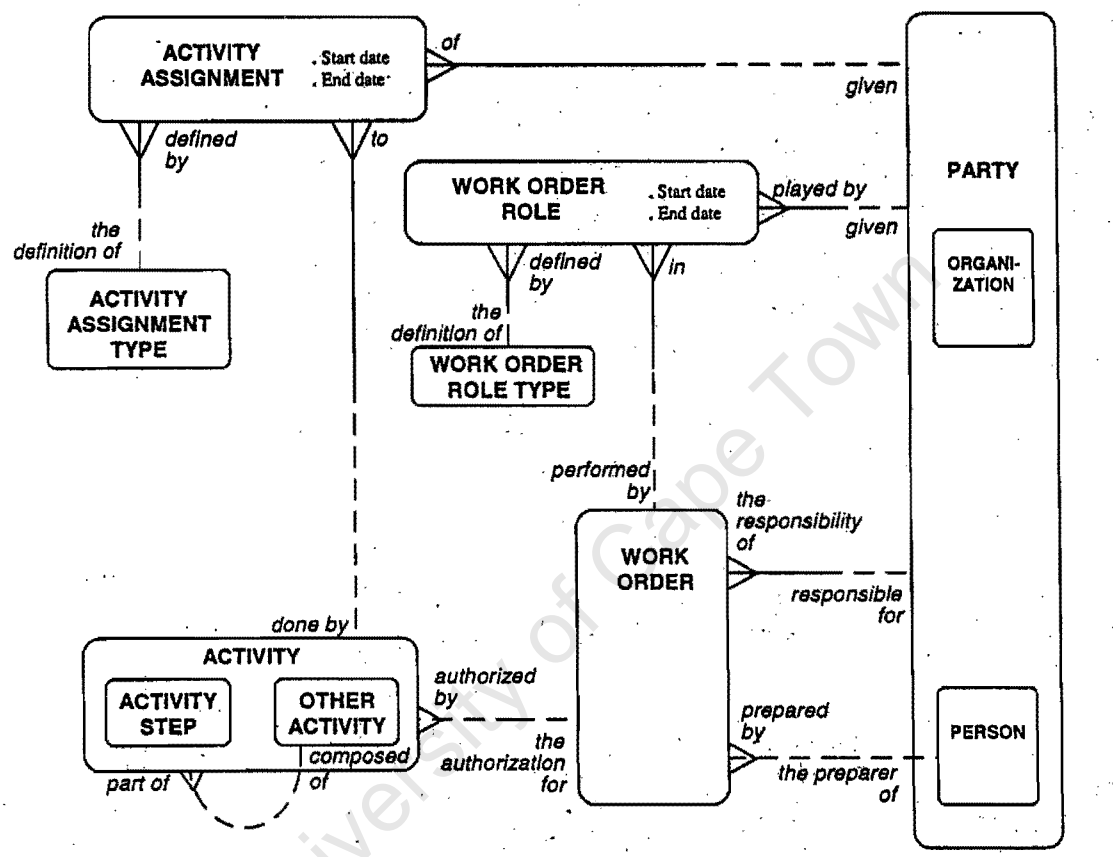
	Orders; 5.5 Roles and Assignments; 5.6 Time Sheets; 5.7 Actual Asset Usage; 5.8 Asset Structures; 5.9 Maintenance Orders; 5.10 Production Orders; 5.11 Production Orders and Lots; 5.13 Dependence; 5.14 Loss Events; 6.1 Purchase Orders; 6.2 Sales Orders; 6.3 Contacts; 6.4 Products and Services; 6.5 Kinds of Contracts; 6.6 Catalogue Item Types; 6.7 User Specifications; 6.9 Control Role Entity; 6.10 Employment Contracts; 6.11 Marketing Responsibilities; 6.12 Deliveries; 6.13 Deliveries of Services; 6.14 Material Movements; 7.1 Accounts; 7.2 Accounting Transactions; 7.3 Revenue; 7.4 Receipts; 7.5 Cash Transactions; 7.6 Expenses; 7.7 External Investment; 7.8 Internal Investment; 7.9 Investment in Labour; 7.10 Depreciation; 7.11 Asset Usage; 7.12 Labour Usage; 7.13 Work Order Completion; 7.14 Cost of Goods Sold; 7.16 Accounting Transaction Types; 7.17 Accounting Rule Entries; 7.19 Allocating Expenses; 7.20 Attributing Revenue; 7.21 Account Categories and Structure ; 7.22 Budgets; 9.3 The manufacturing Model--Reworked; 9.4 The Planning Model; 10.2 Assets and the Process Plant; 10.3 Structure and Fluid Paths; 10.4 Flows; 10.5 Process Entities; 10.6 Actual Processes; 10.7 Conditions and Settings; 10.8 Conditions, Tags, and Measurements; 10.9 The Laboratory and Tags; 10.10 Variable Usage; 11.1 Documents and Copies; 11.4 Document Type Structure; 11.5 Authorship; 11.6 Distribution; 11.7 Other Roles; 11.8 Topics and Index Entries; 11.9 Documenting Products; 11.10 Subject Matter; 11.11 More Subjects; 11.12 Versions; 11.13 Visits and Observations; 11.14 Case report Forms; 11.16 Material Safety Data Sheet Definitions; 11.17 MSDS Sections; 11.18 The MSDS and Asset Types; 11.19 Parameters
Meta-model mapping	
Entity	Type
Name	Type Name
InternalCode	
DefinitionOrDescription	
AttributeName	Attribute Name
Relationship	Relation
Name	Relation Name
InternalCode	
DefinitionOrDescription	
FromEntity	Relationship node
FromRoleName	Relationship role name
FromCardinality	Relationship node crowfoot symbol
ToEntity	Relationship node
ToRoleName	Relationship role name
ToCardinality	(Relationship node crowfoot symbol & dashed/solid line)
RelationshipType	optional {or} mandatory
Generalisation	Super/Sub Type <Containment within box>
Unmapped concepts	<constraints over relationships e.g. OR / AND>

The data models in [HAY96] are in the CASE*Method EERD notation. The following is a sample diagram for Work Orders.

The capture of entities and relationships from the diagrams is extremely straightforward. The following are some examples.

"Activity" is a super-type of "Activity Step" and "Other Activity". "Activity Assignment" has two key attributes: "Start date" and "End date".

There is a relationship between "Work Order" [1..*] and "Party" [0..1] with roles of "the responsibility of" and "responsible for" respectively. There is also a relationship between "Work Order" [0..*] and "Person" [0..1] with roles of "prepared by" and "the preparer of". The "*" cardinality for the "Work Order" entity differs in both cases : a dashed relationship line indicates an optional relationship i.e. 0 cardinality, a solid line indicates a

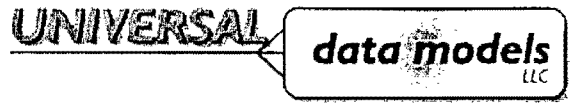


mandatory relationship i.e. a cardinality of 1 (or more). Many diagrams also include non-dashed [1..1] cardinalities.

The diagrams were found to be very correct. Only two minor discrepancies were found:

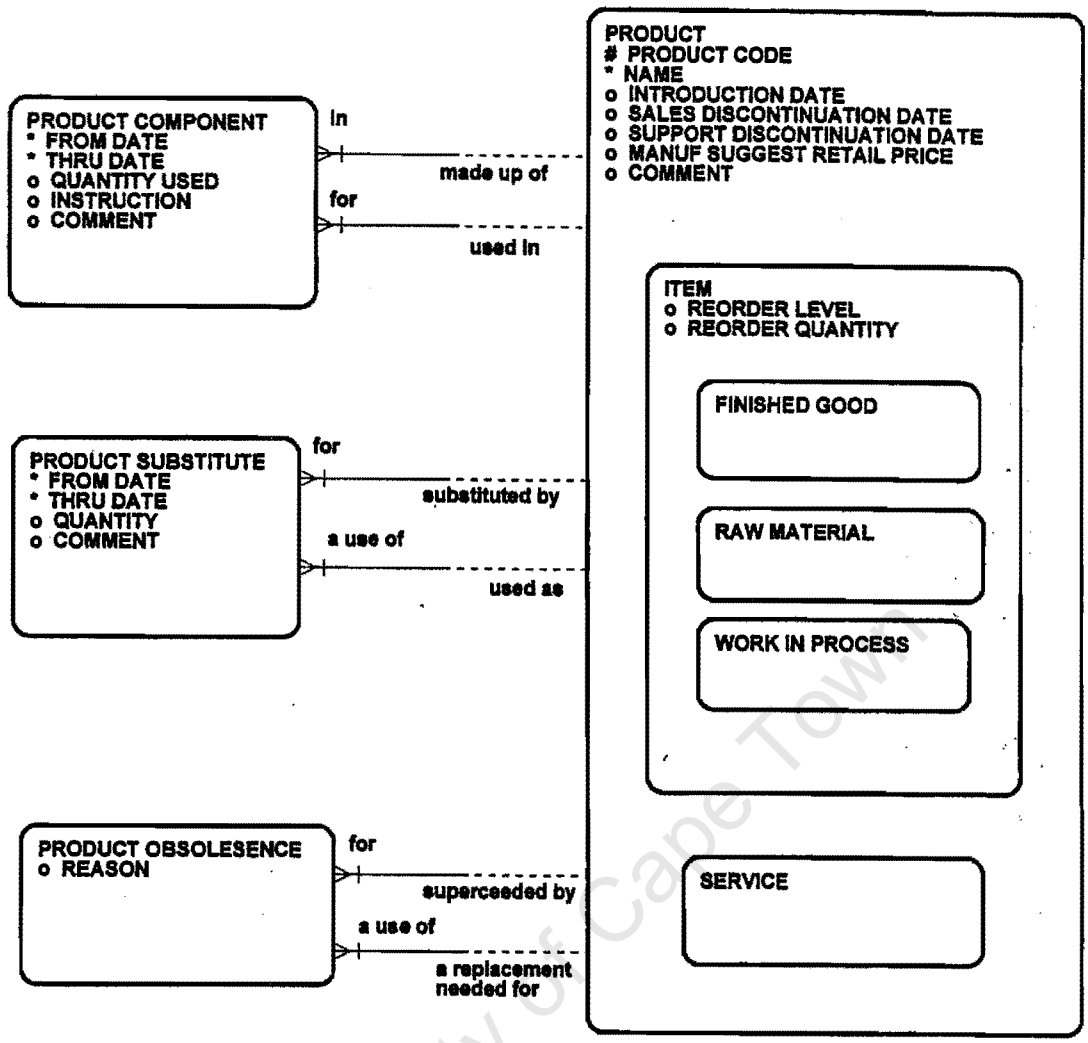
- 10.7 "Conditions/Setting" versus 10.5 "Conditions"
- 10.9 "Material" should be "Material type" as used in all previous diagrams.

A.13 Silverston et al's Universal Data Models

Model ID	Silverston
Model Name	Silverston et al's Universal Data Models
Logo	
Model Description	[SILV97; SILV01] provide a large library of common data models and data warehouse designs for common functions, as well as the rationale behind many of the base 'universal data model' constructs: "Let's face it, most data models are made up of common constructs that have been developed countless times before in other organizations." The first edition was published in 1997. An update (mainly the addition of a chapter) appeared in 2001 as "Volume 1" and provides common data models and data warehouse designs that are useful across industries. An additional Volume 2 provides additional data models applicable to specific industries. The 2001 edition became available only after the model had been captured already, so the models used in this thesis are the ones from the 1997 edition. The book consists of a comprehensive set of detailed models along with instructions to convert the logical data models into enterprise-wide data warehouses and data marts (the "Inmon" influence).
Author / Copyright owner	Silverston, Len; Inmon W.H. & Graziano, Kent
Date (last change)	2001
Primary source type	Printed diagrams in book.
Primary source reference	[SILV97] Silverston, Len; Inmon W.H. & Graziano, Kent. The Data Model Resource Book, A Library of Universal Data Models For All Enterprises. J. Wiley Computer Publishing, New York, 1st Edition.
Secondary sources	[SILV01] Silverston, Len; Inmon W.H. & Graziano, Kent. The Data Model Resource Book, Revised Edition, Volume 1, A Library of Universal Data Models For All Enterprises. J. Wiley Computer Publishing, New York, 2001. Also available in electronic format on CD-ROM from J.Wiley: contains all the diagrams and SQL statements to generate the data structure. See also: http://www.universaldatasolutions.com/
Reference discipline	Data modelling
Modelling notation	EERD notation: CASE*Method by R. Barker, I Palmer & H Ellis. Entities = rounded boxes containing name and attributes; Sub-types = boxes contained in boxes; Relationship = connector line. Dotted = optional / Continuous = Mandatory; Conventional crowsfeet symbols for cardinalities; Role names provided as labels
No. of concepts	267 (of which 196 with detailed attribute listing)
No. of relationships (excl. generalisation)	322
Directed graph?	No
No. of generalisation relationships	82
Depth of inheritance tree	4
Multiple inheritance?	No
No. of grouper constructs	7 : 55
No. of levels in grouping hierarchy	3
Highest level groupings	Top Level: 2. People & Organizations; 3. Products; 4. Ordering Products; 5. Order Delivery and Invoicing; 6. Work Effort; 7. Accounting and Budgeting; 8. Human Resources Second level: 2.3 Party definition; 2.4 Party relationship; 2.6 Address

	definition; 2.7 Contact mechanism definition; 2.8 Contact information; 3.1 Basic product information; 3.2 Product supplier; 3.3 Inventory item storage; 3.4 Standard product pricing; 3.5 Estimated product cost; 3.6 Product components; 4.1 Standard order model; 4.10 Person roles for requests and quotes; 4.11 Agreement definition; 4.12 Relationship of agreement to order; 4.13 Agreement pricing; 4.2 Order definition; 4.3 Order header; 4.4 Order line item; 4.5 Order relationships to party location; 4.6 Person roles for orders; 4.7 Requisition definition; 4.8 Request definition; 4.9 Quote definition; 5.1 Shipment definition; 5.2 Shipment method and shipment vehicle; 5.3 Shipping lots; 5.4 Shipment and order association; 5.5 Invoice definition; 5.6 Shipment and invoice association; 5.7 Invoice billing; 6.1 Work order definition; 6.10 Work task type requirements; 6.11 Work effort invoicing; 6.2 Work order roles; 6.3 Work effort generation; 6.4 Work task definition; 6.5 Work effort and party allocation; 6.6 Work task assignments; 6.7 Inventory assignments; 6.8 Fixed asset assignment; 6.9 Party asset assignments; 7.1 Charts of accounts for internal organisations; 7.2 Accounting transaction definitions; 7.3 Asset depreciation; 7.4 Budget definition; 7.5 Use of budgeted money; 7.6 Budget relationship to general ledger; 8.1 Position definition; 8.2 Position type definition; 8.3 Position reporting relationships; 8.4 Position fulfilment; 8.5 Salary determination and history; 8.6 Benefits definition and tracking; 8.7 Payroll information
Meta-model mapping	
Entity	Type
Name	Type Name
InternalCode	
DefinitionOrDescription	
AttributeName	Attribute Name
Relationship	Relation
Name	Relation Name
InternalCode	
DefinitionOrDescription	
FromEntity	Relationship node
FromRoleName	Relationship role name
FromCardinality	Relationship node crowfoot symbol
ToEntity	Relationship node
ToRoleName	Relationship role name
ToCardinality	(Relationship node crowfoot symbol & dashed/solid line)
RelationshipType	optional {or} mandatory
Generalisation	Super/Sub Type <Containment within box>
Unmapped concepts	<constraints over relationships e.g. OR / AND>

The data models in [SILV97] are also in the CASE*Method EERD notation. All comments as given for Hay's Data Models apply here as well and will not be repeated. The following is a sample diagram.



Silverston *et al* use a slightly different crowfoot to indicate [1..*] cardinality. Also, many more attributes are found in Silverston's diagrams. The Silverston model, true to their co-author Inmon's "data warehousing" background, also includes a full listing of attributes for all entities, along with their type, field length and whether they are mandatory fields or not.

Silverston *et al* also have an explicit set of naming conventions as follows:

Part of name	Meaning
ID	System-generated sequential unique numeric identifier (I.e., 1, 2, 3, 4, …)
seq	System-generated sequence within a parent ID (e.g., order line item sequence number) "
code	Unique pneumonic-used to identify user-defined unique identifers which mav have some meaning embedded in the key (I.e. state code = "CO")
name	A proper pronoun such as a person, geographical area, organization
description	The definition of a unique code or identifier
flag	A binary choice for values (i.e. yes or no; male or female)
from date	Attribute specifies the beginning date of a date range and is inclusive of the date specified
thru date	Attribute specifies the end date of a date range and is inclusive of the date specified (to date is not used since thru date more clearly represents an inclusive end of date range) "

A.14 AKMA's Generic DataFrame

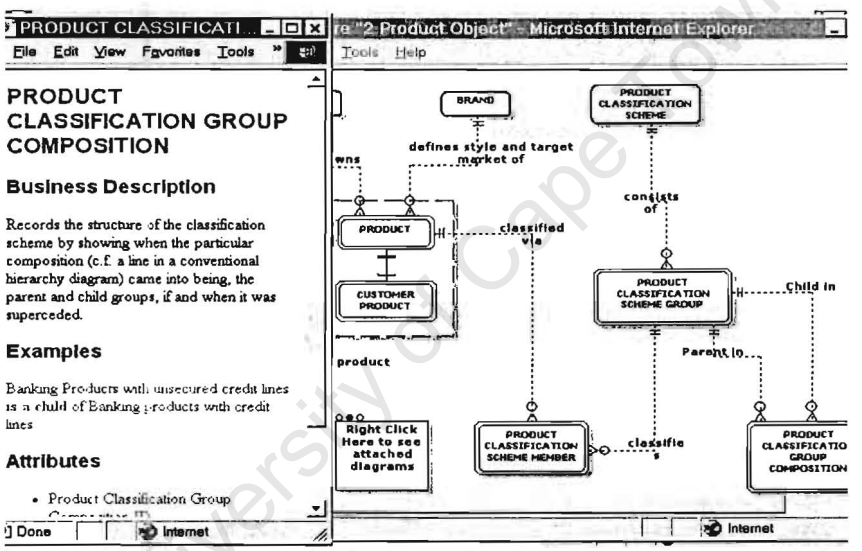
Model ID	AKMA
Model Name	AKMA's Generic DataFrame
Logo	AKMA DataFrames
Model Description	AKMA's Generic DataFrame© is a generic model that exploits the fact that the main information needs of a business in a given sector, or even across sectors, are very similar. The DataFrame is intended as both an operational model and an informational model. That is, it helps with the understanding of the operational information systems and also the historical information needs of decision makers and knowledge workers that are crucial to the implementation of strategic change within an organisation. It is a formal model, generated using the Popkin Software CASE/modelling tool Systems Architecture 2001. Furthermore it is very generic which means that it can be applied easily in many situations and "objects" may be reused throughout the model. The high-level conceptual model that outlines the key concepts and philosophy of the DataFrame approach is publicly available from the AKMA website. It contains those high level objects that relate to major subject areas in the full model as well as the first layer of objects from the Generic Model to allow prospective clients to get a feel for what the full DataFrame provides.
Author / Copyright owner	AKMA Ltd, Network House, 9 Rivers Street Place, Julian Road, Bath, BA1 2RS, UK
Date (last change)	2000
Primary source type	Diagrams, available on-line
Primary source reference	http://www.akma.com
Secondary sources	
Reference discipline	Data modelling / CASE
Modelling notation	EERD notation: System Architect 2001 notation (Popkin Software). Very similar to Hay / Silverston et al.
No. of concepts	82 entities (80 definitions); 383 attributes; 93 examples
No. of relationships (excl. generalisation)	61
Directed graph?	No
No. of generalisation relationships	33
Depth of inheritance tree	2
Multiple inheritance?	No
No. of grouper constructs	6
No. of levels in grouping hierarchy	2
Highest level groupings	1 Party; 2 Product; 3 Delivery Channel; 4 Transactions & Events; 5 Market; 8 Financial Indicators
Meta-model mapping	
Entity	Entity
Name	Entity Name
InternalCode	
DefinitionOrDescription	Business Description
AttributeName	Attribute Name
Relationship	Relation
Name	Relation Name
InternalCode	
DefinitionOrDescription	
FromEntity	Relationship node
FromRoleName	Relationship role name
FromCardinality	Relationship node crowfoot symbol

ToEntity	Relationship node
ToRoleName	
ToCardinality	Relationship node crowfoot symbol & dashed/solid line
RelationshipType	
Generalisation	Super/Sub Type
Unmapped concepts	<constraints over relationships e.g. OR / AND>

The model was downloaded from the web. The web pages contain diagrams, produced by System Architect 2001 using with an EERD notation similar to that of the Hay or Silverston et al models. In addition, each diagram is accompanied by a detailed textual description including

- Business Description: a definition of the concept as well as pertinent notes re the use of the concept e.g. in a data warehouse context.
- Purpose: which explains why an entity is modeled as such.
- Examples: real world examples.
- Definitions used: which other entities and attributes the entity inherits or relates to.
- Attributes: a listing of attributes.

A sample diagram and accompanying textual description as viewed from a browser, is shown below.



The only difference with the EERD notation is the way in which super-types are modeled. In SA2001, these are explicitly modeled using a UML-like relationship, but with a short thick line perpendicular to the relationship line at the parent node (instead of an open arrowhead).

Some notes re the capturing process.

- There is some inconsistency in the naming of relationships (i.e. some have an initial capital, but most do not), this inconsistency was preserved during the capturing.
- A number of classes were undefined:
  - E080 TRANSFER
  - E081 POST CODE CLASSIFICATION GROUP COMPOSITION
- A number of hyperlinks failed to link to an existing HTML file, but in most cases the proper file name could be guessed and loaded.
- A number of classes have no attributes.
- All defined classes have a description
- Many classes have examples

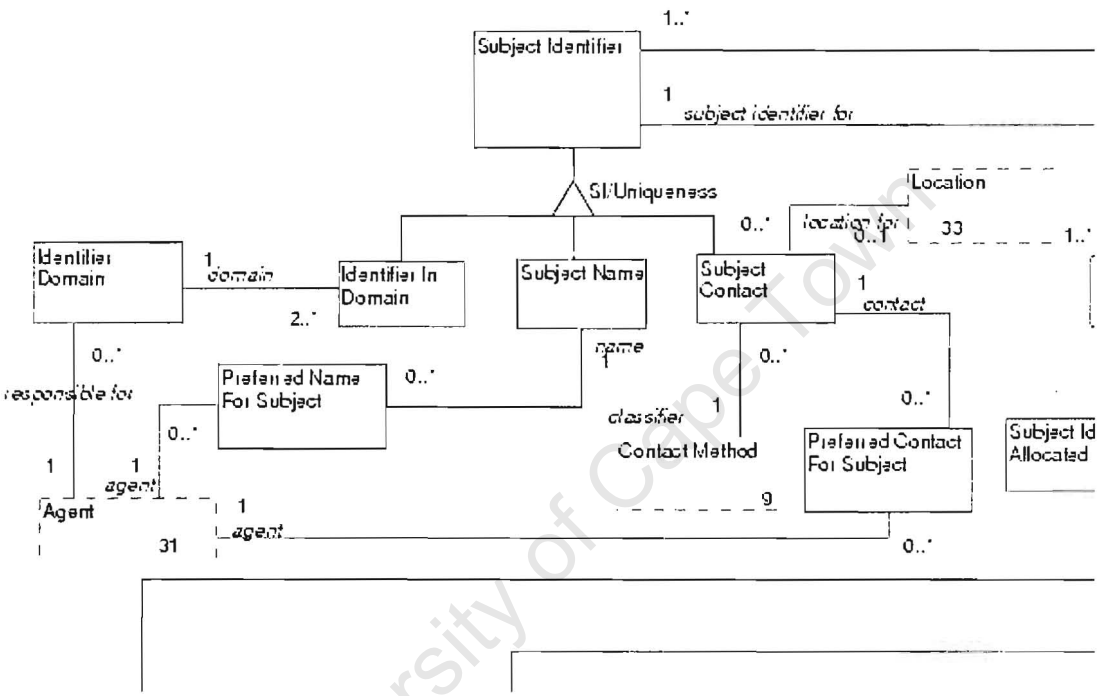
Relationships are only named in one direction: generally from the [0..1] or [1..1] to the [0/1..M] direction.

A.15 NHS Generic HCM Class Model

Model ID	NHS
Model Name	NHS Generic HCM Class Model
Logo	NHS Information Authority
Model Description	The models on the NHS web site represents their understanding of the provision of patient care in the English and Welsh National Health Service. They are information models a`nd their purpose is to describe a healthcare system in the real world. One of the models (Provide Patient Care) reflects the real world processes carried out in a healthcare system. This includes the delivery of care to patients, the management of the delivery of care and associated resources, and links to the underlying external knowledge, terminology and classification systems. The Generic Class model describes the types of data (or objects) which underpin the Provide Patient Care model, and a lot more besides. Only the Generic Class model was used for this research. Its advantage is that the domain appears generic enough yet it is a good example of a more industry specific model.
Author / Copyright owner	National Health Services (UK)
Date (last change)	1999
Primary source type	A large diagram, available on-line as a PDF file.
Primary source reference	A text file containing a repository dump from CASE tool. http://www.standards.nhsia.nhs.uk/hcm
Secondary sources	Also available on CD-ROM on written request from: NHS Information Authority", Aqueous II, Aston Cross, Rocky Lane, Birmingham, B6 5RQ, U.K. Turn-around time was less than two weeks. (Copy received on 6 June 2001 from "Amanda".)
Reference discipline	System Engineering
Modelling notation	UML v.1.3 Class diagrams & Formatted text
No. of concepts	269 concepts (216 with definitions; 53 specialisations); 50 attributes; 11 examples
No. of relationships (excl. generalisation)	220 (of which 17 are duplicates)
Directed graph?	Yes
No. of generalisation relationships	236
Depth of inheritance tree	6
Multiple inheritance?	No
No. of grouper constructs	
No. of levels in grouping hierarchy	1
Highest level groupings	
Meta-model mapping	
Entity	Type
Name	##pf1##
InternalCode	
DefinitionOrDescription	##pf3## <following> ##pf1##
AttributeName	##pf2##Attributes:
Relationship	Association
Name	##pf5## <name>
InternalCode	
DefinitionOrDescription	
FromEntity	##pf2##Associations From
FromRoleName	

FromCardinality	##pf2##Associations From: <cardinality>
ToEntity	##pf2##Associations To
ToRoleName	
ToCardinality	##pf2##Associations To: <cardinality>
RelationshipType	
Generalisation	##pf2##Subtype Partitions: <or> ##pf2##Supertypes:
Unmapped concepts	

The NHS web site makes the Generic HCM Class model of the "Provide Patient Care" system available. This "Health Care Model" is also available on CD-ROM. The model consists of both a graphic version and a formatted text version. The formatted text version was used, since a lot of the data capture could happen automatically.



The model comes as a single large diagram in PDF format. However, a CASE repository "dump" in formatted text was also provided. This was used as the basis for the model capture.

The following gives an extract of the formatted text for the concept "activity". ##pf1##Activity identifies the key concept "Activity"; the next concept defined is "Activity For Subject".

##pf1##Activity
##pf3##Purposeful and intentional "Event"
##pf4##Previous documented definitions providing the source for the above are: * An exertion of energy initiated by an "Agent" (CBS v2) - this definition is followed by a definition of its subtypes. * Conscious, directed and purposeful exertion of energy. * "Event" occurring through the volition of an "Agent".
##pf2##Attributes:
##pf3##Quantity (M) (Number)
##pf4##Quantity of an Activity.
##pf2##Associations From:
##pf5##Only One activity for One or Many Activity State
##pf5##Only One replacement activity Zero, One or Many Activity Substitution
##pf2##Associations To:
##pf5##Zero, One or Many classifier Only One Activity Type
##pf5##Zero, One or Many unit for Only One Unit Of Measure
##pf5##Zero, One or Many urgency classifier Zero or One Urgency
##pf2##Supertypes:
##pf4##Event
##pf2##Subtype Partitions:
##pf3##Activity/Direction
##pf4##Activity For Subject

##pf4##Temporal Resource Act ##pf4##State Change Activity ##pf3##Activity/Repetition ##pf4##Single Activity ##pf4##Repetitive Activity ##pf1##Activity For Subject

This is following by the definition: ##pf3## identifies the first line and ##pf4## the continuation of the definition.

The heading "##pf2##Attributes:" precedes a list of attributes (here only one), namely "Quantity" (of the type "Number", whose description is found on the line identified by ##pf4##.

The heading "##pf2##Associations From:" precedes the list of relationships which depart from this entity. Each "##pf5##" line identifies, in order: from cardinality, relationship name, to cardinality and target entity.

The heading "##pf2##Associations To:" precedes the list of relationships which arrive at this entity. These are the target nodes of the "##pf2##Assocations To" heading under the corresponding entity.

Finallly, the headings "##pf2##Subtype Partitions:" and "##pf2##Supertypes:" precede lists of subtypes and supertypes respectively. Although these are also supposed to be symmetrical or corresponding, the "supertypes" equivalent was not found where the subtypes were not listed or defined entities (see note below).

Some notes re the capture:

- A "symmetry"-check for relationships was possible: for every "Associations From" underneath the domain concept, there should be a corresponding "Associations To" entry for the target concept. This was the case in all but two concepts.
- A similar check was made for the "Supertypes" and corresponding "Subtype Partitions" entries. There was full correspondence between the respective entries. However, tt was found that a large number of subtype partitions were listed where the subtype was not a listed entity e.g. for "Activity" above, neither "Activity/Direction" nor "Activity/Repetion" are listed entities. These were included into the database and added another 53 (specialization) entities to the 216 defined entities.
- 17 relationships repeat themselves - with changing node cardinalities. These were excluded from the database. Two examples are:

From Entity	From Cardinality	Relationship Name	To Cardinality	To Entity
Timepoint	Zero or One	expected restart	Zero, One or Many	Performance Suspended Activity
Timepoint	Only One	expected restart	Zero, One or Many	Performance Suspended Activity
Subject	Only One	subject	Zero, One or Many	Object Representation
Subject	Only One	subject	One or Many	Object Representation

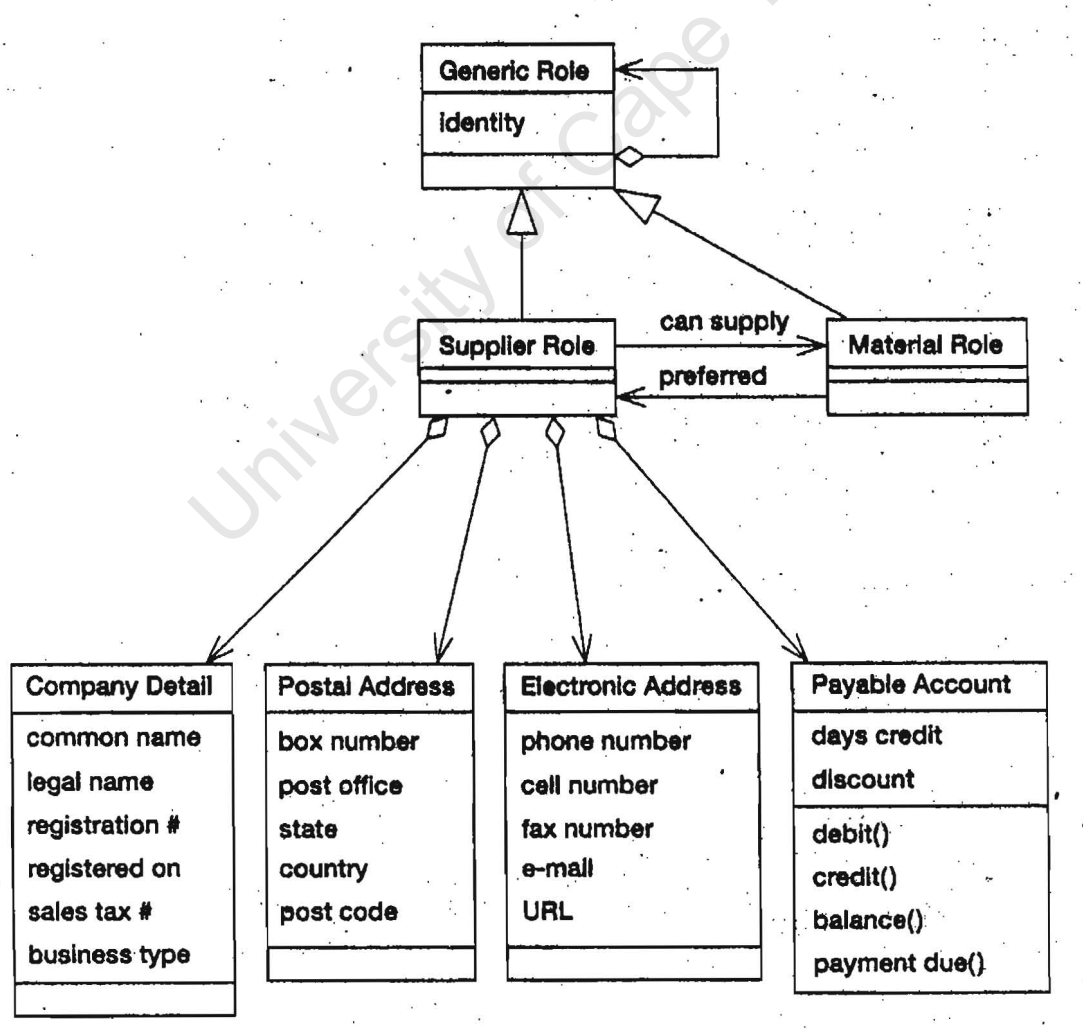
A.16 BOMA

Model ID	BOMA
Model Name	Marshall's BOMA Model
Logo	
Model Description	Chris Marshall describes a specific way in which to model enterprises from a business rather than a technical perspective. The book contains a number of high-level, conceptual models, very much at the level of Fowler's analysis patterns, but from a fully object-oriented perspective. His meta-model is based on the key business object types: entity, process, purpose / organization as endorsed by the OMG. It is not a coincidence that he is closely associated with its Business Object domain task force. The book not only includes the diagrams and a very readable description of / motivation for the models, but it also includes an appendix in which his models are applied to a case study. In addition, the BOMA CD-ROM included with the book provides an electronic copy of the diagrams as well as the necessary JAVA code to generate and/or customize the classes.
Author / Copyright owner	Chris Marshall, SESH Holdings
Date (last change)	2000
Primary source type	Printed diagrams in book.
Primary source reference	[MARS00] Marshall, Chris. Enterprise Modelling with UML. Designing Successful Software Through Business Analysis. Addison-Wesley: Reading (MA), 2000.
Secondary sources	Re-engineer from the JAVA source code contained on the accompanying CD-ROM.
Reference discipline	Business Objects
Modelling notation	UML v.1.3 Class diagrams JAVA classes
No. of concepts	183
No. of relationships (excl. generalisation)	192 (125 aggregation; 2 composition; 30 dependency; 25 navigation; 10 association)
Directed graph?	Mostly
No. of generalisation relationships	115
Depth of inheritance tree	5
Multiple inheritance?	No
No. of grouper constructs	4 : 38
No. of levels in grouping hierarchy	3
Highest level groupings	Highest level: Entity; Process; Organization; Example Second level: Account Values; Actor Roles; Actors and Roles; Artifact Values; Asset Artifact; Cash Sale; Claim Process; Contract Manager; Customer Party; Customer Role; Employee Party; Employee Role; Entity Roles; Entity Values; Finance Artifact; Finance Process; Financial Ledger; Human Resource; Inventory Options; Machine Resource; Material Role; Message Manager; Money Role; Organization Unit; Party Roles and Values; Payment Process; Process Manager; Product Artifacts; Product Role; Production Process; Production Schedule; Purchase Process; Roles and Values; Sales Process; Space Resource; Supplier Role; Value-Adding Process; Work in Process.
Meta-model mapping	
Entity	Class
Name	Class Name
InternalCode	

DefinitionOrDescription	(most entities have a formal description captured from the CD-ROM)
AttributeName	attribute
Relationship	association {or} aggregation {or} composition {or} navigation
Name	association or navigation label
InternalCode	
DefinitionOrDescription	
FromEntity	From {or} Child {or} Part
FromRoleName	
FromCardinality	
ToEntity	To {or} Parent {or} Whole
ToRoleName	
ToCardinality	
RelationshipType	association {or} aggregation {or} composition {or} navigation
Generalisation	generalisation
Unmapped concepts	operation()

Initially, an attempt was made to capture the model directly from the JAVA source code on the BOMA CD-ROM , but the inter-class relationships were difficult to identify. These had to be captured from the accompanying (more general) book, which contained also many classes that did not appear in the JAVA model. Also, the JAVA model had quite a few "technical implementation" classes e.g. 11 printer-related classes (out of 135). It was then decided to abandon the BOMA model and use the models from the book instead.

The following gives a typical example of one of the UML models. Marshall makes generous use of aggregation, composition and inheritance relationships.



The capture of entities and relationships from the UML model is straightforward. Figure 1-4 on page 9 of the book explains the subset of UML notations as used in the book.

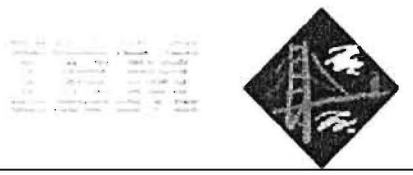
Some minor inconsistencies were encountered during the capturing process: e.g. in the chapter dealing with the "process objects", most process entities have their main attributes and methods indicated; this is not the case in the other chapters/entities. For the non-process business objects, very few relationships are named, mainly where possible confusion arises e.g. relationships between message header and message manager are "new, sent, waiting, and received".

The appendix lists a worked out example of how to use the generic constructs and, since this added additional generic relations, these were also included in the captured model

The following relationship types were used (together with their "intuitive" equivalent)

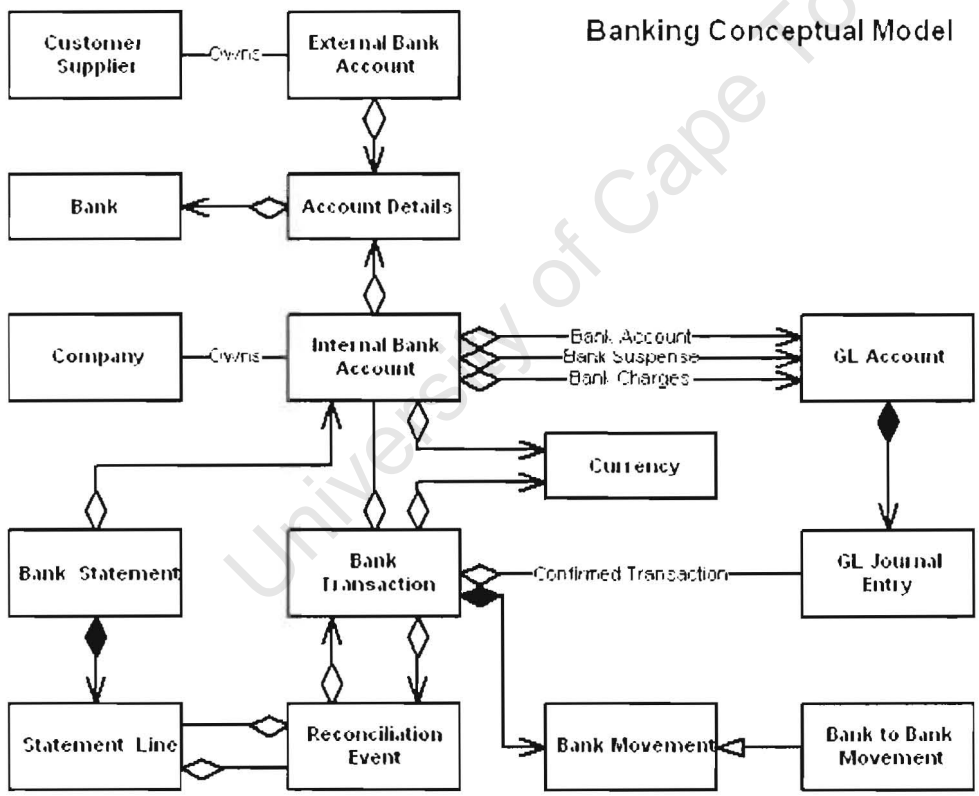
- Generalisation = is-a(-subtype of)
- Relation = association
- Aggregation = contains
- Composition = consists of
- Navigation = association with direction
- Dependency = depends on

A.17 IBM's San Francisco Application Business Components

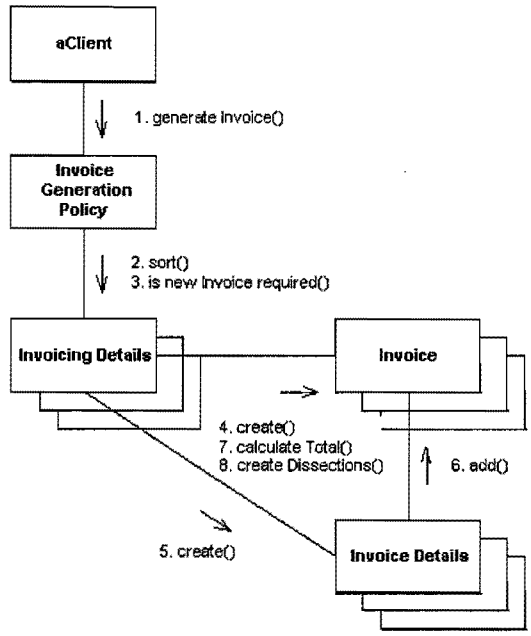
Model ID	SanFran
Model Name	San Francisco Application Business Components
Logo	
Model Description	The San Francisco Framework is a commercial, OO application framework. It consists of a set of generic, high-level JAVA classes representing business entities. These classes cooperate in a user-defined way to implement core business processes. It is one of the first examples of a framework to include high-level business functionality, instead of the usual low-level system focus. IBM is now no longer marketing IBM SanFrancisco™ V2.1; it is superceded by WebSphere Business Components Studio Version 1.2 as from 11September, 2001.The underlying idea is that flexible yet robust business components can be built for multi-platform systems with a minimum of complexity, expense and time investment The underlying philosophies are those of object-orientation, portability, client/server technology, platform independence, sound architectural principles and RAD.
Author / Copyright owner	IBM
Date (last change)	2000 (version 2.1)
Primary source type	Diagrams from the San Francisco Framework documentation.
Primary source reference	http://www-4.ibm.com/software/ad/sanfrancisco/ Re-engineer from the JAVA source code contained in the San Fancisco Framework package.
Secondary sources	See also: [BENN00] Ben-Natan, Ron & Sasson, Ori. IBM SanFrancisco: Developer's Guide. McGraw-Hill, New York, 2000.
Reference discipline	Business Objects
Modelling notation	UML v.1.3 Class diagrams & Interaction diagrams
No. of concepts	109
No. of relationships (excl. generalisation)	172 (52 aggregation; 47 composition; 73 association)
Directed graph?	No
No. of generalisation relationships	11
Depth of inheritance tree	3
Multiple inheritance?	No
No. of grouper constructs	40
No. of levels in grouping hierarchy	2
Highest level groupings	Address Implementation; Add Transaction To Financial Batch; Banking Conceptual Model; BP Balances; Business Partner; Business Partner View; Business Partner View Addr; Company Hierarchy; ContactManagement; CreateFinancialBatch; CreateGenericBankMovement; CreateGenericBankTransaction; CreateInvoice; CreditCheck; CreditCheckConfigCon; CreditCheckPolicy; CreditCheckPolicyOrg; CurrencyGailLossAccounts; CurrencyRequirements; ExchangeRateCollection; ExchangeRates; FiscalCalendar; GenericOrderDescription; GenericOrderIntegration; GenericOrderLineItemDescription; InitialsAbilityGroup; InitialsExtension; InstallmentValueCalculationo.esult; Invoice; Measurement; NaturalCalendar; NaturalCalendarwithWorkPeriods; PartyManagement; PaymentPlan; PaymentTerms; QuantityCon; SuppChargeCon; TransactionEuro; TransactionValueRequires; UnitCategoryCon
Meta-model mapping	
Entity	Class

Name	Class Name
InternalCode	
DefinitionOrDescription	
AttributeName	Attribute
Relationship	Association {or} aggregation {or} composition {or} operation()
Name	Association or navigation label
InternalCode	
DefinitionOrDescription	
FromEntity	From {or} Child {or} Part
FromRoleName	
FromCardinality	
ToEntity	To {or} Parent {or} Whole
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	
Unmapped concepts	

The entire package can be downloaded from the IBM website. As for the BOMA model, an attempt was initially made to re-engineer the JAVA code provided, but abandoned for the same reason. It was decided to make use of the documentation for the "CBOs" (Common Business Objects). This documentation is in HTML form (text and GIFs) and takes different formats e.g. both UML structure & interaction diagrams. It appears that both are complementary.



The following are two sample diagrams. The interpretation and model capture from the UML class diagram is straightforward.



Some of the UML interaction diagrams added entities not covered in the UML class diagrams and these were also captured. Interactions (method invocations) were also modeled as relations, except that different methods invoked between the same classes were captured as a single relationship i.e. there would be a single (directed) relationship from "Invoicing Details" to "Invoice" with the name "create; calculate total; create dissections". The motivation is that these are often part of the same context & transaction.

Some noteworthy capturing notes:

- The documentation was clearly produced by a number of different people using slightly different standards. This also resulted in a number of inconsistencies between diagrams. An example is "PartyRole" which is probably the same as "Business Partner Role" Also "QuantityUnit" is elsewhere named "Quantity Unit".
- Where very obvious duplications occur between diagrams (e.g. one diagram to create, another to delete financial batches), these "duplications" have been omitted. Where they occured in distinct diagrams, they were kept.
- There were quite a few spelling errors e.g. in "...CurrencyGailLossAccounts.gif" : "UrealizeLoss" Posting Combination.
- Not all differently named entities were captured e.g. Credit Check Policy, D(efault)Credit Check Policy and your Credit Check Policy all are instances of the same class (the last two are assumed to be special instances of generic credit check?)
- Quite a few ""Dxx""to""xx"" relationships were (and D – for *Default* – entities) omitted.
- There are a number of inconsistencies in naming e.g.: "PaymentMethod" but "Payment Plan" and "Payment Detail"

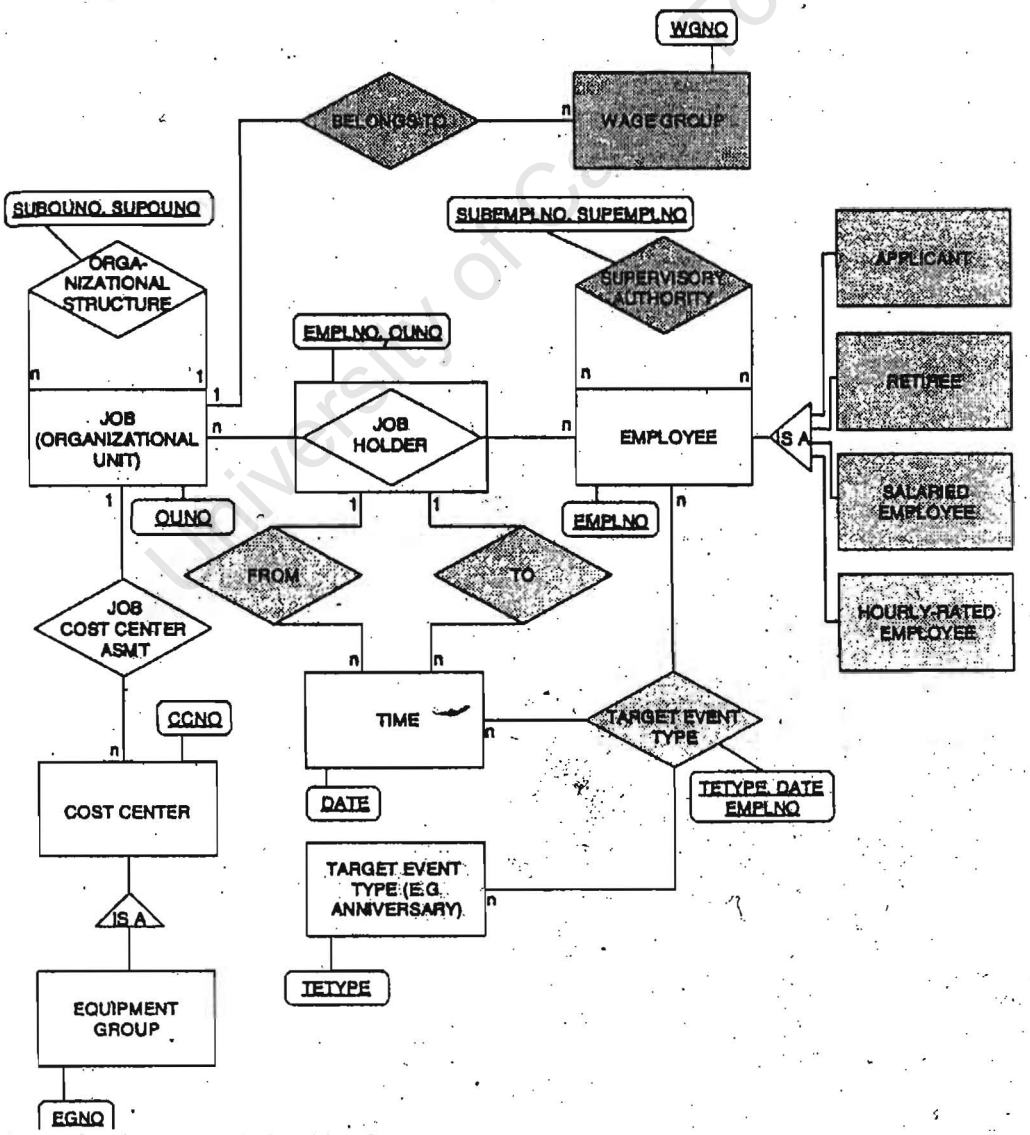
A.18 SAP R/3's Reference Model

Model ID	SAP
Model Name	The SAP R/3 Reference Model
Logo	
Model Description	SAP R/3 is by far the most successful ERP system on the market and the perfect example of how a generic model can apply to organisations across a multitude of industries; although the extensive customization and implementation processes illustrate equally well the gap that remains. The SAP R/3 Reference Model is based on the research done by Prof Scheer. In [SCHE98], he published the underlying "Reference Models for Industrial Enterprises". This is an extensively researched and well-documented 770 page "model" for general enterprise models. The ground work for the models was published in a first edition in 1989. This, in turn, was based on the research done as part of the "Cologne Integration Model" (KIM). The book contains 580 figures but, luckily, there is a much smaller number of "summary diagrams". Although the book contains both process as well as data models, only the latter have been captured (as explained in the methodology section). It must be noted that the SAP ERP system relies heavily on the process models; also its underlying data model has developed quite substantially from its roots, not in the least due to its move to object-orientation and e-commerce integration. The model was captured from the 24 summary data diagrams.
Author / Copyright owner	Prof A-W Scheer; SAP AG
Date (last change)	1994
Primary source type	Printed diagrams in book.
Primary source reference	[SCHE98] Scheer, August-Wilhelm. Business Process Engineering. Reference Models for Industrial Enterprises. Springer-Verlag, Berlin, 1998 (2nd ed.).
Secondary sources	The model should correspond at least partially with the repository dump of SAP Business Object Repository ("BOR") or viewable by means of the ARIS modelling tool.
Reference discipline	Enterprise Resource Planning
Modelling notation	EERDs.
No. of concepts	404
No. of relationships (excl. generalisation)	612
Directed graph?	No
No. of generalisation relationships	160 (of which 34 are multiple inheritance)
Depth of inheritance tree	4
Multiple inheritance?	Yes
No. of grouper constructs	17
No. of levels in grouping hierarchy	3 (only 2 captured)
Highest level groupings	Requirements planning; scheduling and capacity planning; production; inbound logistics; purchase order handling; outbound logistics; sales; human resource management; personnel accounting; human resource planning; planning; design engineering; document administration; product data; financial accounting; cost accounting; workflow management
Meta-model mapping	
Entity	Entity type {or} relationship type
Name	Entity Name

InternalCode	
DefinitionOrDescription	
AttributeName	Key field
Relationship	Relationship
Name	Relationship name
InternalCode	
DefinitionOrDescription	
FromEntity	Entity node
FromRoleName	
FromCardinality	Node cardinality
ToEntity	Entity node
ToRoleName	
ToCardinality	Node cardinality
RelationshipType	Relationship type
Generalisation	IS-A
Unmapped concepts	

Although the book contains both process as well as data models, only the latter have been captured (as explained in the methodology section).

The following (summary) diagrams from the book were captured: B.I.66; B.I.105; B.I.119; B.I.221; B.II.07; B.II.10; B.II.21; B.II.25; B.III.05; B.III.09; B.III.13; B.IV.02; C.II.11; C.II.33; C.II.54; D.I.03; D.I.16; D.I.18; D.I.19; D.I.22; D.I.24; D.I.30; D.II.16; D.II.18. Below is an example of one of the smaller diagrams, figure B.III.05: the basic data structure for human resource management.



The capture of entities and relationships from the above diagram is straightforward. Entity capture is facilitated even further by Scheer's convention to have entities that are newly introduced (i.e. not appearing in previous diagrams) appear in light-grey boxes. Note the difference between a ("proper") entity such as EMPLOYEE and

a relationship entity such as JOB HOLDER. Not all N-to-N relationships were normalized into relationship entities. Relationship entities were only created by SCHEER when the relationship participated in further relationship. Where a relationship carried its own attributes (i.e. became a relational table) but did not participate in other relationships, it was modeled as a diamond. We thus have three types of entities: pure entities (as present in the domain) modelled as rectangles, normalized M:N relationships with attributes (diamond inside rectangles) and relationships with attributes (diamonds).

Reflecting its relational database background, the diagram also shows primary key-fields for most entities e.g. EGNU for EQUIPMENT GROUP. Note the "IS-A" generalization from EQUIPMENT GROUP to COST CENTER

The following capturing notes are relevant.

- For Fig B.I.221 CAM model, no cardinalities given but many could be inferred from the symbols e.g. N:N relationships.
- All N-to-N relationships were captured as entities but not the N-to-1 relationships, except if the relationship was already entitized elsewhere
- Interesting is the frequent use of multiple inheritance of which there are 34 (out of 160!) cases. For example Plant is a subtype of Business Area and Organizational Unit. Often this is done for *technical* reason, namely merely to inherit the relevant attributes, for example, Inspection Plan is a subtype of Routing, Customer Terms and Supplier Terms

A.19 Baan's DEM Business Reference Model

Model ID	BAAN
Model Name	The Baan IV DEM Business Reference Model
Logo	
Model Description	BAAN Company is one of the leading ERP solution providers worldwide. Its current product is the BAAN (release) IV system. The Baan system is based on a 1985 software package that included finance, manufacturing and distribution modules. This package was enhanced, using an MRP II approach, in the 1989 "Triton Software", which is really the first release of the ERP system. In 1990, the system was moved to a client/server environment and during 1993 and 1994 the process, transportation, project control and EIS modules were added. The reference model underlying the system has, as far as known, not been made available in any publicly available document, although the model can be accessed easily by any license holder through the Orgware Dynamic Enterprise Modeler (DEM) tool. Not only would this probably infringe intellectual property rights, it also clashes with the initial methodology (public availability to ensure repeatibility and verification by other researchers). Hence a decision was taken to "reverse engineer" the model underlying BAAN IV by using the publicly available book that best describes its structure. After some research, it was clear that the tables and screenshots in [PERR98] were sufficiently detailed and the book covered a sufficiently large area of the BAAN IV system.
Author / Copyright owner	BAAN
Date (last change)	1997
Primary source type	Printed tables and screenshots in book.
Primary source reference	[PERR98] Perreault, Yves & Vlasic, Tom. Implementing Baan IV. Que, Indianapolis (IN), 1998.
Secondary sources	The model should correspond at least partially with the repository dump of a Baan system or viewable by means of the DEMO modelling tool.
Reference discipline	Enterprise Resource Planning
Modelling notation	Relational database tables & screenshots
No. of concepts	322
No. of relationships (excl. generalisation)	608
Directed graph?	Yes
No. of generalisation relationships	142
Depth of inheritance tree	4
Multiple inheritance?	No
No. of grouper constructs	8
No. of levels in grouping hierarchy	3 (only 2 captured)
Highest level groupings	Modules: Common; Finance; Manufacturing; Distribution; Inventory; Transportation; Process; Service
Meta-model mapping	
Entity	Table {or} screen
Name	(Name)
InternalCode	
DefinitionOrDescription	(Description from text)
AttributeName	Field
Relationship	Relationship
Name	Relationship name
InternalCode	
DefinitionOrDescription	

FromEntity	Entity node
FromRoleName	
FromCardinality	
ToEntity	Entity node
ToRoleName	
ToCardinality	(1:N - see notes)
RelationshipType	
Generalisation	
Unmapped concepts	(see notes)

The BAAN IV reference model had to be re-engineered from the published description. This obviously introduced a number of errors and omissions, e.g. generalization relationships were likely to be lost. The book that was used for the re-engineering effort, is the one with the most extensive description of the BAAN IV system.

It is written by two different authors. The authors, each apparently responsible for those modules they know best, appear to have adopted somewhat different approaches in their description of BAAN IV. The one uses screenshots, the other lists detailed tables with attributes and attribute descriptions. Both were used as inputs to capture the model, as described below. It must be noted that a number of BAAN modules are dealt with very summarily and it can only be assumed that the relevant reengineered sections of the model are deficient when compared to the true underlying model. The reengineered model is really that of the relational database underlying the implement Baan IV system.

The following is an illustrative example of how the tables were reengineered into entities and relationships.

Table 18.3 Sales Order Lines Fields

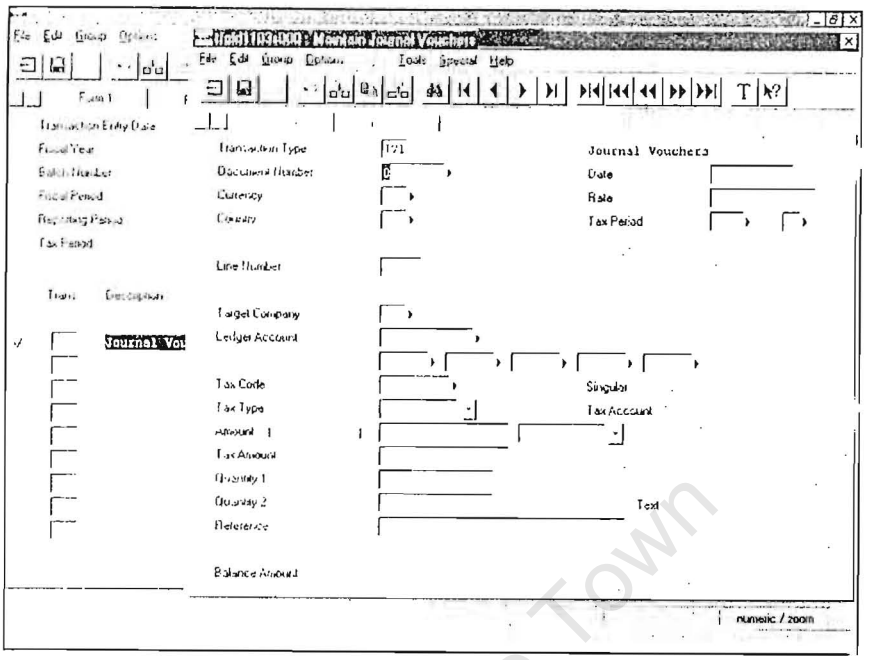
Field Name	Description
Sales Order	Order number.
Position Number	Line number.
Project	Project number.
Item Code System	Customer profile code for system containing customer/company item cross-reference.
Item	Company part number.
Container	Company container number. (A container is part of a set.)
Item Description	Company part description.
Item Category	Indicates if an item is standard or customized.
Engineering Item Revision	Code for controlling manufactured parts.
Product Variant	Base item used for configuring a customized item.
Warehouse	Ship from warehouse.
Delivery Date	Date order will ship.
Text	Indicates whether order line has a message attached.
Sales Unit	The unit in which the item is shipped.
Quantity	The quantity requested by the customer for delivery.

continues

The table obviously implies the creation of the entity "Sales Order Lines". From the field names, it was inferred (assumed?) that there is a relationship with the following entities: Sales Order, Project, Item Code System, Item, Item Category, Container, Engineering Item, Product Variant and Warehouse. Most of these are entities defined elsewhere in the book. The fields Position Number, Delivery Date, Tex and Quantity are attributes of the entity "Sales Order Lines" whereas Item Description and Sales Unit are attributes of some of the entities mentioned in the relations.

A lot of the reengineering was based on semantic interpretation and general understanding of business domain. This is obviously subjective and a number of errors are likely to have resulted from the process, although the overall general "appearance" of the model should not be affected dramatically.

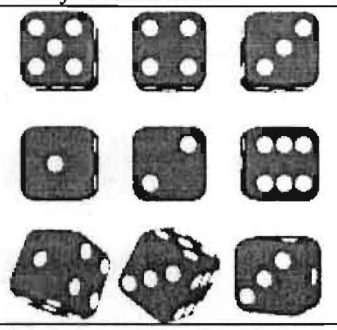
The screen shots were easier to reengineer. On screen shots, the triangle ▶ indicates a selection list, which can be interpreted as a 1:N relationship to another table. In tables no such clear link exists so 1:N relationships are derived by the name of the attribute.



The following are some general reengineering and capturing notes.

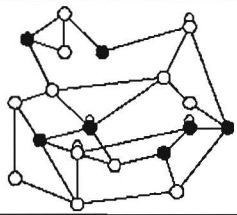
- Since this is a flat relational model, no generalization relationships are mentioned explicitly. Some could be derived from the text (e.g. "Types of warehouses"); others from a ▼ sign on the screenshots.
- The highest level groupings are "modules". These are subdivided into smaller, lower-level modules, down to the "business object" level. All entities and relationships were grouped into the highest level module since there is some vagueness in which lower level most entities belong.
- There are a number of slight inconsistencies: e.g. "line of business" p. 145 is singular whereas all other tables are plurals - and even in text it is called "lines of business" p.155.
- Fig 14.2 was incorrectly reproduced: it is not the *financial customer group* but *match invoice with order*.
- The concept of a business object in BAAN IV seems to be that of a low level module, not that of an entity.
- The first couple of modules have been described in detail and hence fairly accurate reengineering was possible. However, for some of the later modules there are no tables or screenshots, so I had to rely on fairly vague textual descriptions which complicated the reengineering effort esp. for manufacturing & production management sections. Note that in actual implementations not all modules would normally be implemented e.g. one would configure BAAN IV into either assemble-to-order OR engineer-to-order OR project industry.
- Sometimes spurious relationships are introduced. For example, in BAAN some items can be specified at the customer, sales order or sales order line level e.g. currency, sales contract and tax code. This introduces 3 relationships whereas it should really be modelled at the lowest level only - the higher levels are "defaults". This is even more the case for specifying e.g. discounts, which can be done at 8 different levels! However, this corresponds to the model used in SAP R/3.

A.20 The Random Model

Model ID	RANDOM
Model Name	A fully random model
Logo	
Model Description	This is intended to be a fully random model i.e. a model without semantic content, containing 258 randomly selected English words (the concepts/entities) and 455 relationships between them. The numbers are designed to match the size of the semantically richest "OttawaDense" model below. It is designed in such a way that each concept contains at least one link. This model is called "Random1". The words were selected from the Oxford Paperback Dictionary [OXFO79] by selecting the column head words on approximately each third page. Where the headword was not a noun (or where it was a proper noun or double word), the next column would be selected. Because the dictionary is just short of 800 pages, the necessary extra nouns were added at appropriate relative intervals by skipping just two pages instead of three. 455 pairs of random numbers were generated. These accounted for the 455 random relationships, with each pair of numbers corresponding to a "from" and "to" entity. To create random generalizations (sub/supertype relationships), the average ratio of generalisation relationships to concepts/entities in all the captured models (generic model database) was calculated. This ratio, 70.1%, was applied to the number of entities in the random model, to yield a desired number of 320. In fact, 322 random generalization relationships had to be created, since 2 were reflexive i.e. entity numbers 168 & 246 had themselves as their own supertype! No grouper constructs were created. Note that random model could have been made even more random by using nonsensical random character strings instead of existing English words. There may be a bias in the headword selection since words with multiple meanings are more likely to be included than single meaning terms (the former occupy more dictionary space). This should not impact the analysis.
Author / Copyright owner	Jean-Paul Van Belle
Date (last change)	2002
Primary source type	Entities from English word list.
Primary source reference	Relationships from (pseudo-)random numbers.
Secondary sources	Entities: [OXFO79] The Oxford Paperback Dictionary. Oxford University Press, Oxford, 1979.
Reference discipline	Relationships: MS-Excel random number generator =RAND( )
Modelling notation	
No. of concepts	
No. of relationships (excl. generalisation)	
Directed graph?	258
No. of generalisation relationships	455
Depth of inheritance tree	Yes
Multiple inheritance?	320
No. of grouper constructs	15
No. of levels in grouping hierarchy	Yes
Highest level groupings	

Meta-model mapping	
Entity	
Name	
InternalCode	Entity
DefinitionOrDescription	Entity Name
AttributeName	
Relationship	
Name	
InternalCode	Relationship
DefinitionOrDescription	Relationship Name
FromEntity	
FromRoleName	
FromCardinality	From Entity
ToEntity	
ToRoleName	
ToCardinality	To Entity
RelationshipType	
Generalisation	Generalisation
Unmapped concepts	

A.21 The Semi-Random Model

Model ID	SEMI-RANDOM
Model Name	A model with business entities and random relationships
Logo	
Model Description	This is intended to be a semi-random model, with entities related to the domain (the organisation) but with meaningless relationships between the entities. It contains the 258 English business words selected from the OttawaDense model, but with (the same) 455 random relations between them as in the Random model. The same inheritance structure was also maintained.
Author / Copyright owner	Jean-Paul Van Belle
Date (last change)	2002
Primary source type	Entities from English word list. Relationships from (pseudo-)random numbers.
Primary source reference	Enities: The Oxford Paperback Dictionary Relationships: MS-Excel random number generator =RAND( )
Secondary sources	
Reference discipline	
Modelling notation	
No. of concepts	258
No. of relationships (excl. generalisation)	455
Directed graph?	Yes
No. of generalisation relationships	320
Depth of inheritance tree	15
Multiple inheritance?	Yes
No. of grouper constructs	
No. of levels in grouping hierarchy	
Highest level groupings	
Meta-model mapping	
Entity	Entity
Name	Entity Name
InternalCode	
DefinitionOrDescription	
AttributeName	
Relationship	Relationship
Name	Relationship Name
InternalCode	
DefinitionOrDescription	
FromEntity	From Entity
FromRoleName	
FromCardinality	
ToEntity	To Entity
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	Generalisation
Unmapped concepts	

A.22 Ottawa-Big

Model ID	OTTAWA-BIG
Model Name	Ottawa's Business Dictionary Hyperlinks Model
Logo	Ottawa Business Journal
Model Description	This is a semantically based "hyperlink" version of a business terms dictionary i.e. a "model" consisting of domain-specific concepts with relationships representing semantic references in the lexicon definitions of the concepts. The source lexicon was the "Business Dictionary" available on the website of the Ottawa Business Journal as available on 17 November, 2001. It contains just over 900 entries with definitions which frequently contain references to other terms in the dictionary, indicated by capitalization. These references can be thought of as hyperlinks though they are not implemented as such in the online source lexicon. After deleting a few spurious key terms (e.g. R&D: See RESEARCH AND DEVELOPMENT) and correcting some of the spelling mistakes, 901 terms with definitions were retained. In these definitions, a total of 1050 references was made, of which only 634 could be identified as referring to one of the 901 terms. The relationships linked only a subset of 459 terms, implying 442 "orphan" entities i.e. terms that do not refer to, and are not mentioned by, any other terms in the lexicon. The 459 entities, with the 634 relationships linking them, form the "OttawaBig" model. Of these terms, only 201 participate in one single relationship.
Author / Copyright owner	The Ottawa Business Journal
Date (last change)	2001
Primary source type	Dictionary (semantic network)
Primary source reference	http://www.ottawabusinessjournal.com/business_tools/business_dictionary/ The Business Dictionary Tool of the Ottawa Business Journal website.
Secondary sources	
Reference discipline	Linguistics
Modelling notation	Natural language descriptions
No. of concepts	901 / 459
No. of relationships (excl. generalisation)	634
Directed graph?	Yes
No. of generalisation relationships	
Depth of inheritance tree	
Multiple inheritance?	
No. of grouper constructs	
No. of levels in grouping hierarchy	
Highest level groupings	
Meta-model mapping	
Entity	{Business term)
Name	(Word in CAPITAL letters)
InternalCode	
DefinitionOrDescription	(Business Dictionary Description)
AttributeName	
Relationship	(Word referred to in CAPITALS in entity definition)
Name	
InternalCode	
DefinitionOrDescription	
FromEntity	(Defined entity)

FromRoleName	
FromCardinality	
ToEntity	(Defining or referred entity)
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	
Unmapped concepts	Non-capitalized terms

A.23 Ottawa-Dense

Model ID	OTTAWA-DENSE
Model Name	Ottawa's Business Dictionary Denser Hyperlinks Model
Logo	 Ottawa Business Journal
Model Description	The "Ottawa Dense Model" is a smaller, more compact version of the Ottawa Business Dictionary Semantic Model. That model contained 901 terms, of which only 459 were linked. Of these terms, 201 only participate in one single relationship. In order to create a more compact model with a denser network of relationships, these terms with the relationship linking them were deleted to form this second model, consisting of 258 entities and 455 relationships. Some relationships linked 2 "single-relationship" terms, so less than 201 relationships were removed from OttawaBig.
Author / Copyright owner	The Ottawa Business Journal
Date (last change)	2001
Primary source type	Dictionary (semantic network)
Primary source reference	http://www.ottawabusinessjournal.com/business_tools/business_dictionary/
Secondary sources	
Reference discipline	Linguistics
Modelling notation	Natural language descriptions
No. of concepts	258
No. of relationships (excl. generalisation)	455
Directed graph?	Yes
No. of generalisation relationships	
Depth of inheritance tree	
Multiple inheritance?	
No. of grouper constructs	
No. of levels in grouping hierarchy	
Highest level groupings	
Meta-model mapping	
Entity	{Business term)
Name	(Word in CAPITAL letters)
InternalCode	
DefinitionOrDescription	(Business Dictionary Description)
AttributeName	
Relationship	(Word referred to in CAPITALS in entity definition)
Name	
InternalCode	
DefinitionOrDescription	
FromEntity	(Defined entity)
FromRoleName	
FromCardinality	
ToEntity	(Defining or referred entity)
ToRoleName	
ToCardinality	
RelationshipType	
Generalisation	
Unmapped concepts	Non-capitalized terms

APPENDIX B: CLASSICAL SYSTEM ENGINEERING METRICS APPLICABLE TO MODELS

B.1 Cyclomatic complexity [EDMO99; SHEP95]

Of the 48 syntactic complexity measures investigated by Edmond, the cyclomatic complexity measure was found to be the one that is the most suitable for model analysis. In its original definition, the cyclomatic complexity of a graph is the number of independent loops in that graph. It can be calculated by the formula:

$$\boxed{\text{Cyclomatic Complexity} = CC = R - E + P}$$

With

- R = number of arcs (relationships),
- E = number of vertices or nodes (entities) and
- P = number of disjoint partitions the graph divides into

Cyclomatic complexity is meant to capture the *inter-connectedness* of a model. A pure tree-structure will have R = E-1 (each entity has exactly one parent, except the top-most node; there is no multiple inheritance) and hence CC = (E-1) - E + 1 = 0 whereas a complex interrelated model with many loops will have a much higher value. There is no direct relationship between the size of the model as measured by the number of entities and the CC.

In a modelling context, relationships can include or exclude inheritance relationships and/or grouping relationships.

McGabe used CC to measure program complexity by interpreting m as the number of discrete logical paths (decision points) the code execution could follow (with p = 1). This was meant to be indicative of the number of tests needed as well as measuring the difficulty of maintaining the program. There are a number of criticisms on McGabe's use of CC, but most of them relate to the re-interpretation from arcs/relations to logical paths/decision points.

Studies indicate a strong correlation between CC and the number of errors existing in source code. McGabe found that a cyclomatic complexity of 10 appeared to be a practical upper limit for module size. This criterion is clearly untenable for models where the number of relationships often exceeds the number of entities by a large number.

B.2 Kolewe's Association Complexity [HEND96]

By reverse interpretation of McGabe's CC, Kolewe proposes the association complexity metric AC as follows:

$$\boxed{\text{Association Complexity} = AC = R - E + 2P}$$

Note the close correspondence between AC and CC. No explanation could be found for the difference between the two formulae (P as opposed to 2xP). Note that Kolewe's AC excludes inheritance and grouping relationships explicitly.

B.3 Yi and Winchester's graph impurity measure [SHEP95]

The graph impurity measure is computed similarly to the cyclomatic complexity but using module calls (relationships) outside their regular modules hierarchy (entity inheritance tree). A high value is interpreted the same way as a high module fan-out (see below).

B.4 Absolute and Relative Connectivity

Apart from CC, [EDMO99] cites the total number of relations as a measure for connectivity i.e. the extent of inter-connections between model components. Since this is heavily influenced by the model size, one can also calculate the number of relations relative to the number of entities.

$$\boxed{\text{Absolute Connectivity} = AC = R}$$

$$\boxed{\text{Relative Connectivity} = RC = R / E}$$

The connectivity measures are meant to gauge model complexity.

B.5 Kitchenham's module fan-out [SHEP95]/ Chidamer & Kemerer's design fan-out [HEND96]

Kitchenham's module fan-out measures the number of subordinate modules for a module. A high value is indicative of a missing level of abstraction. Several measures can be calculated for a module: highest and average module fan-out for the model and, orthogonally, module fan-out based on inheritance tree or directed relationships.

Chidamer & Kemerer's "CBO" (Coupling Between Object classes) version is calculated at the class level and refers to the number of other classes referenced. Special versions exist for "friend classes" and distinctions can be made according to the type of reference [PRES97c].

An important consideration is not just the average fan-out value across all classes, but also the distribution (e.g. standard deviation) of the fan-out values. "For a system in which a single class has a very high fan-out and all the other classes have low or zero fan-outs, we really have a structured, not an OO system." [HEND96]

B.6 DeMarco's Data Bang [SHEP95]

DeMarco suggested two bang metrics: one for function-strong and another for data-strong metrics. For data models, only the Data Bang metric is applicable and can be calculated as follows:

$$\text{Data Bang} = DB = \sum COBI_i = \sum RE_i \times dw_i$$

With

- RE_i = the count of relationships of the i-th entity
- dw_i = the weighting factor obtained from a table supplied by DeMarco (see below)
- $COBI_i$ = the corrected object increment, calculated as the product of RE_i and dw_i for each entity, effectively the contribution of the model entity to the overall bang measure for the entire data model.

The following table lists the Data Bang metric function complexity weighting factors and resultant COBIs.

RE (Relationship count)	dw (weight)	COBI (corrected object increment)
1	1.00	1.0
2	1.15	2.3
3	1.33	4.0
4	1.45	5.8
5	1.56	7.8
6	1.63	9.8

Although DeMarco suggests that the metric is 'quantitative indicator of net usable function from the user's point of view', Sheppard disagrees and suggests that additional empirical research evidence is required. MacDonell finds slightly more merit in the measure when comparing it to others, but agrees on the lack of validation [MACD94]

B.7 Class Count [HEND96]

The simplest size measure is the total number of classes in a system or model. The most common, popular, well-known and widely used metric in programming is LOC - lines of code. Its equivalent in the modelling world is probably the class count (although CASE size may come close). Henderson-Sellers agrees that "this is an extremely rough measure, but one can say that a system with 1000 classes in it is likely to be bigger in all senses of the word) than one with only 20 classes."

B.8 CASE Size or Concept Count [MACD94]

CASE size is a slightly more sophisticated version of the class count. CASE size measures the number of entries in the CASE tool's data dictionary i.e. it is a count of all entities, attributes and relationships. Although this metric is influenced by the meta-model used and the amount of detail captured, it remains very useful for comparative purpose and actually came out "tops" in MacDonell's comparison of complexity measures.

$$\boxed{CS = E + A + R}$$

With

- E = entity count
- R = relationship count
- A = attribute count

It is not specified whether grouping constructs (groupers and grouping relations) are meant to be included. Since these are stored separately in the data dictionary, there is a strong argument to include them. Although CS is an immediate size count, it turns out to be a fairly good proxy for complexity as well.

B.9 Fenton's Morphology Metrics for the Inheritance Graph

Fenton [FENT96] suggested the following rather simple morphology metrics to summarize the shape of the model inheritance graph (tree).

Graph Size = Entities Count + Relationships Count

Connectivity Density = Arc-to-Node Ratio = Relationships Count / Entities Count

Depth = Longest path from root (top) node to a leaf node

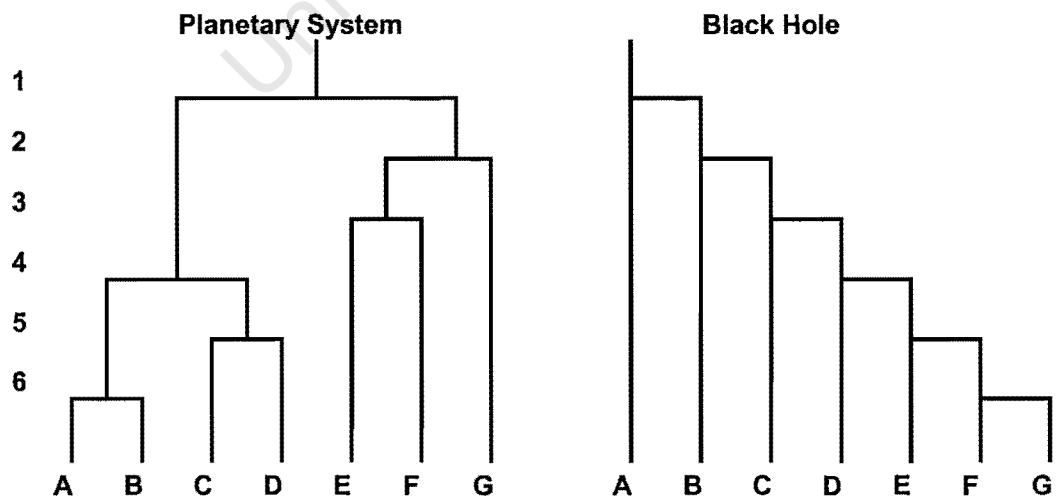
Width = Maxium number of nodes at any one level of the architecture

The first two can be extended to include all possible relationships, in which case the Graph Size become synonym with Case Size (excluding attributes) and the Connectivity Density equals the Relative Connectivity ratio. The depth is equal to Lorentz & Kidd's class hierarchy nesting level.

B.10 Cluster analysis.

A number of cluster measures exist, all having to do with the grouping of similar modules. Similarity is typically based on the reduction of a matrix showing the inter-module (class?) relationships/couplings. A standard approach is to count the number of calls between modules and to cluster modules with high counts, often using the nearest neighbour-algorithm: start with clustering the two modules with the highest count and combining their rows / columns, and repeating the process until a certain minimum cut-off value is reached or all modules are grouped into one.

An interesting by-product of the sequential grouping process is the dendogram which is a graphic illustration of the module-by-module grouping process. This provides a "fingerprint" of the clustering process which is characteristic of the model or system. Typical dendogram types are "planetary" and "black hole" dendograms as per examples below. However, it must be remembered that the dendogram shape is dependent on the choice of clustering algorithm.



In the context of a model, the procedure is not very useful since few entity pairs share more than one relationship between them. However, the use of dendogram and cluster analysis proved very useful for measuring the similarity and degree of overlap between models.

B.11 Martin's Dependency Coupling Metrics [MART95]

[MART95] covers a number of interesting additional dependency metrics, of which the following are of specific interest:

- Afferent coupling: the number of classes from other (subject) categories that depend on classes within a subject category.
- Efferent coupling: the number of classes in other categories that the classes in the subject category depend on.
- Instability: ratio of efferent coupling to total coupling.

To apply these metrics to models requires the identification within the model of both dependency relationships and category groupings. This is possible for models where directed relations have been identified and by taking subject areas or individual diagrams as the category groupings. Where directed relations are missing, non-directed relationship could possibly be interpreted as bi-directional relationships.

B.12 Lorenz & Kidd's Class Inheritance Metrics [LORE94]

Most of the OO design metrics discussed/proposed by Lorenz & Kidd require access to the class internals e.g. class methods, variables. The only ones applicable to the high level used in enterprise models are the ones relating to class inheritance structure. This group of metrics looks at the quality of the classes' use of inheritance.

B.12.1 Class hierarchy nesting level.

The class hierarchy nesting level is a count of the maximum number of levels in the inheritance hierarchy. As Lorenz & Kidd point out, this is likely to be higher when OO frameworks or libraries are used. They suggest that the maximum threshold should be six (from the top of the class hierarchy or the bottom of the framework). "Large nesting numbers indicate a design problem, where developers are overly zealous in finding and creating objects. [...] Deep class inheritance nesting is not necessary or desirable. Testing is more difficult and the real world does not typically contain this much specialization." However, if it is too low, the designer did not make adequate use of the benefits offered by the inheritance or specialisation/generalisation mechanism.

[HEND96] calls the maximum class hierarchy nesting level the maximum depth of inheritance tree.

B.12.2 Number / proportion of abstract classes

An abstract class cannot be instantiated but is used to facilitate reuse of methods and attributes or generalises constructs in the domain. Frameworks rely heavily on abstract classes and Kidd & Lorenz claim that "well-designed projects" typically contain 10 to 15 percent abstract classes.

B.12.3 Number of key classes

A related metric proposed by [KIDD94] is the (proportional) number of key classes [NKC] with a key class defined as one that focuses directly on the business domain. Non-key classes are implementation or infrastructure related e.g. GUI, communications etc. The value suggested for this metric is between 20% and 40% for a typical OO system. For the generic enterprise models, however, the value should be as close to 100% as possible, since they are supposed to be pure implementation-independent domain models.

B.12.4 Number of subsystems

Splitting a model up into smaller cohesive submodels assists with the resource allocation, scheduling and overall integration effort. The number of subsystems is indicated by the count NSUB.

B.12.5 Use of multiple inheritance

OO and conceptual modelling in general supports multiple inheritance, and a number of OO languages such as C++ and Smalltalk implement this. However, Lorenz & Kidd, as well as many other practitioners [PRES97c] do not recommend the use of multiple inheritance in modelling businesses and consider it "an anomaly", to be used in exceptional cases only. Some of the complications resulting from multiple inheritance are name collisions when inheriting attributes or methods from superclasses, the cognitive comprehension problems in working through two or more inheritance trees, and the complications in maintaining systems. Lorenz & Kidd put it in the list of metrics because they "don't believe in its use and therefore want to detect its usage." They therefore recommend a threshold value of zero.

This metric is equivalent to what Binder calls module fan-in [FIN] which translates, in an OO model, to the number of classes from which a class inherits its attributes / operations. Therefore FIN > 1 is equivalent to multiple inheritance and it is suggested that this should be avoided.

B.13 Binder's Number of Root Classes [PRES97c]

This metric, NOR, counts the number of distinct class hierarchies. Most of the above inheritance metrics should be calculated for each root class hierarchy but can also be calculated for the model as a whole (including or excluding classes that fall outside of any inheritance hierarchy).

B.14 Henderson-Seller's Inheritance Coupling Measures

[HEND96] introduces a number of additional inheritance-based measures.

B.14.1 Mean depth of inheritance tree [HEND96]

The average depth of the inheritance tree, assuming that all classes belong to a single or number of parallel inheritance trees, is calculated by determining the depth of each class in its hierarchy. It is misleadingly called DIT or Depth in Inheritance Tree, and referred to as the "nesting level" by Lorenz & Kidd, or "class-to-root" depth by Tegarden and Sheetz.

$$\text{Average Inheritance Depth} = \text{AID} = \sum \text{DIP}_i \ / \ \text{total number of classes}$$

B.14.2 Number of children for each class [HEND96]

Each class in the inheritance tree has zero or more children, which is their NOC.

One can calculate the average NOC for the whole inheritance tree as well as the standard distribution and full distribution of NOC for the inheritance tree.

It is possible to include or exclude the leaf classes for both DIT and NOC.

B.14.3 Yap & Henderson-Sellers Class Re-use and Specialisation Metrics [HEND96]

In order to evaluate the extent to which the designer re-used classes with hierarchies, two ratios can be computed.

$$\text{Reuse Ratio} = U = \text{number of superclasses / total number of classes}$$

The reuse ratio indicates the extent to which the designers (or prospective implementers) of the class library have been (will be) able to inherit from their own classes to create new classes. U is always less than one, but gets closer to one when there is a large number of classes and a fairly linear hierarchy, or when multiple inheritance is used a lot. Conversely, a value near 0 indicates a shallow depth.

The second of their ratios to measure re-use is:

$$\text{Specialization Ratio} = S = \text{number of subclasses / number of superclasses}$$

The specialization ratio measures the extent to which a superclass has captured the abstraction. A large S indicates a high degree of reuse since there are lots of subclasses. With multiple inheritance, the value can be less than 1 although generally values of S (and U) close to 1 suggest poor design.

B.14.4 Chunking Metrics [HEND96]

Henderson-Sellers devotes a substantial amount of space to the measurement of cognitive complexity, by means of chunking. Unfortunately his approach is based strongly on programming statements and constructs.

However, most enterprise models fragment the enterprise model by grouping related entities and relationships on single diagrams. These can be interpreted as cognitive chunks and metrics can be computed for them including the distribution and average number of entities, relationships, and specialisations per diagram.

B.15 Sears' Layout Appropriateness [PRES97c]

Sears has proposed layout appropriateness [LA] as a metric for human-computer interfaces in a GUI environment. The metric is based on the existence of layout entities such as icons, menus, windows etc. in a GUI, which the user must navigate in order to achieve certain tasks.

It is possible to re-interpret the metric to measure the LA of model diagrams where the overall task is "semantic interpretation by the user".

In order to calculate the LA of a diagram, one must first calculate the cost of inspecting the existing diagram and relate that to the cost of inspecting the optimal diagram.

$$\text{Cost} = \sum [\text{ frequency (k) x cost (k) }]$$

With

- k = a specific transition from one diagram entity to the next
- frequency (k) the number of times that transition needs to be made to interpret the diagram
- cost (k) the cost associated with making the transition.

Transitions could be from any diagram element to any other diagram element, although there should be some spatial or cognitive cost associated with the transition.

One possible specialization would be to only consider transitions between entities and associating a cost with the transition based on the nature, length and other attributes of the relationship symbol. E.g. cognitive cost of a relationship drawn as a line would increase with the number of bends, crossings and length of each relationship. Presumably horizontal and vertical lines have a lower cost than diagonal lines. A one-to-one relationship would have a lower cost than a one-to-many relationship, which would in turn have a lower cost (weighting) than a many-to-many relationship. A uni-directional relationship is lower in cost than a non- or bi-directional relationship. Pre-defined relationships such as generalisation/specialisation and Is-An-Instance-Of would have a lower semantic cost than semantically defined ones; especially if special and appropriate diagramming forms are use (groupers, containment). Standards such as always diagramming one-to-many relationships for top-left to bottom-right would also reduce cost.

When the costs and frequencies of all transitions have been calculated, it is possible to compute LA as follows.

$$\text{LA} = 100 \text{ x (Cost of LA-optimal Layout) / (Cost of Actual Diagram Layout)}$$

The cost of LA-optimal Layout is the cost of the layout with the lowest possible cost. In principle one would have to generate all possible diagram layouts (which is a factorial function of number of grid positions and the number of diagram elements) and calculate the cost of each of them. In practice, more efficient heuristics, such as tree searching algorithms, can be used.

B.16 IEEE's Software Maturity Index

IEEE standard 982.1 proposes a software maturity index to indicate the maturity/stability of a software product based on the number of changes that have occurred since the previous release.

Software Maturity Index = SMI = $[M_T - (F_a + F_b + F_c)] / M_T$

With

- M_T = the total number of modules in the current release
- F_a = the number of modules in the current release that have been changed
- F_b = the number of modules in the current release that have been added
- F_c = the number of modules in the current release that have been deleted

As M_T starts to approach 1, it indicates a more mature product/model. In the context of a model, the term "module" could be reinterpreted as a diagram or model element.

B.17 Readability Scores for Documentation

The following three metrics measure readability of documentation.

B.17.1 Flesch Reading Ease score

Rates text on a 100-point scale; the higher the score, the easier it is to understand the document. A standard document should score approximately 60 to 70.

$$\text{Flesch Reading Ease} = 206.835 - (1.015 \text{ x ASL}) - (84.6 \text{ x ASW})$$

where:

- ASL = average sentence length (the number of words divided by the number of sentences)
- ASW = average number of syllables per word (the number of syllables divided by the number of words)

B.17.2 Flesch-Kincaid Grade Level score

Rates documentation text on a U.S. grade-school level. A score of 7.0 means that a seventh grader can understand the document. Most standard documents should have a score of approximately 7.0 to 8.0.

$$\text{Flesch-Kincaid Grade Level} = (.39 \times \text{ASL}) + (11.8 \times \text{ASW}) - 15.59$$

where:

- ASL = average sentence length (the number of words divided by the number of sentences)
- ASW = average number of syllables per word (the number of syllables divided by the number of words)

Note: this is incorrectly reprinted as (.39 x ASL) + (100 x ASW) - 15.59 in [GILL97].

B.17.3 Fog Index

The fog index is calculated as

$$\text{Fog Index} = 0.4 \times \text{ASL} + \text{WW2S}$$

Where:

- ASL = average sentence length (the number of words divided by the number of sentences)
- WW2S = the percentage of words with more than two syllables

APPENDIX C: AESTHETIC MEASURES FOR SCREEN LAYOUT

The formulas and descriptions for following measures were taken *verbatim* and summarized as given by Ngo, Teo and Byrne [NGO00], most of which was officially published in [NGO01]. The source should be consulted for more detailed descriptions and the reasoning behind each of the measures. They relied heavily on Galitz's book on design and layout [GALI97] where he presents an extensive list of very specific guidelines for the design of screens, as well as the more conceptual [BIRK33]. All of the measures have been normalised so that the calculated values range from 0 (worst) to 1 (best).

C.1 Measure of Balance

Balance can be defined as the distribution of optical weight in a picture. Balance is computed as the difference between total weighting of components on each side of the horizontal and vertical axis and calculated as follows:

$$BM = 1 - \frac{|BM_{vertical}| + |BM_{horizontal}|}{2} \quad \in [0,1] \tag{1}$$

$BM_{vertical}$ and $BM_{horizontal}$ are, respectively, the vertical and horizontal balances with

$$BM_{vertical} = \frac{w_L - w_R}{\max(|w_L|, |w_R|)} \tag{2}$$

$$BM_{horizontal} = \frac{w_T - w_B}{\max(|w_T|, |w_B|)} \tag{3}$$

with

$$w_j = \sum_i^{n_j} d_{ij}\left(\frac{a_{ij}}{a_{\max}} + |c_{ij} - c_{frame}| + s_{ij}\right) \quad 0 \le \{c_{ij}, c_{frame}, s_{ij}\} \le 1, \quad j = L, R, T, B \tag{4}$$

a_{max} is the area of the largest object on the frame with

$$a_{\max} = \max(a_{ij}, i = 1,2,\ldots,n_j, j = L, R, T, B) \tag{5}$$

where L, R, T, and B stand for left, right, top, and bottom, respectively; a_{ij}, c_{ij}, and s_{ij} are, respectively, the area, colour, and shape of object i on side j; d_{ij} is the distance between the central lines of the object and the frame; and n_j is the total number of objects on the side. Each colour is given a weighting between 0 (white) and 1 (black). Shapes are described using the method given [BIRK33]

C.2 Measure of Equilibrium

Equilibrium is a stabilisation, a midway centre of suspension. Equilibrium on a screen is accomplished through centring the layout itself. It is computed as the difference between the centre of mass of the displayed elements and the physical centre of the screen and is given by

$$EM = 1 - \frac{|EM_x| + |EM_y|}{2} \quad \in [0,1] \tag{6}$$

The equilibrium components along the x-axis (EM_x) and y-axis (EM_y) are given by

$$EM_x = \frac{2\sum_i^n a_i (x_i - x_c)}{nb_{frame}\sum_i^n a_i} \tag{7}$$

$$EM_y = \frac{2\sum_i^n a_i(y_i - y_c)}{nh_{frame}\sum_i^n a_i} \quad (8)$$

where (x_i,y_i) and (x_c,y_c) are the co-ordinates of the centres of object i and the frame; a_i is the area of the object; b_{frame} and h_{frame} are the width and height of the frame; and n is the number of objects on the frame. (Note that the maximum values of $|x_i-x_c|$ and $|y_i-y_c|$ are $b_{frame}/2$ and $h_{frame}/2$.)

C.3 Measure of Symmetry

Symmetry is axial duplication: A unit on one side of the centre line is exactly replicated on the other side. Vertical symmetry refers to the balanced arrangement of equivalent elements about a vertical axis, and horizontal symmetry about a horizontal axis. Radial symmetry consists of equivalent elements balanced about two or more axes that intersect at a central point. Symmetry, by definition, is the extent to which the screen is symmetrical in three directions: vertical, horizontal, and diagonal and is given by

$$SYM = 1 - \frac{|SYM_{vertical}| + |SYM_{horizontal}| + |SYM_{radial}|}{3} \in [0,1] \quad (9)$$

$SYM_{vertical}$, $SYM_{horizontal}$, and SYM_{radial} are, respectively, the vertical, horizontal, and radial symmetries with

$$SYM_{vertical} = \frac{\begin{array}{l}|X'_{UL} - X'_{UR}| + |X'_{LL} - X'_{LR}| + |Y'_{UL} - Y'_{UR}| + |Y'_{LL} - Y'_{LR}| + \\ |H'_{UL} - H'_{UR}| + |H'_{LL} - H'_{LR}| + |B'_{UL} - B'_{UR}| + |B'_{LL} - B'_{LR}| + \\ |\Theta'_{UL} - \Theta'_{UR}| + |\Theta'_{LL} - \Theta'_{LR}| + |R'_{UL} - R'_{UR}| + |R'_{LL} - R'_{LR}|\end{array}}{12} \quad (10)$$

$$SYM_{horizontal} = \frac{\begin{array}{l}|X'_{UL} - X'_{LL}| + |X'_{UR} - X'_{LR}| + |Y'_{UL} - Y'_{LL}| + |Y'_{UR} - Y'_{LR}| + \\ |H'_{UL} - H'_{LL}| + |H'_{UR} - H'_{LR}| + |B'_{UL} - B'_{LL}| + |B'_{UR} - B'_{LR}| + \\ |\Theta'_{UL} - \Theta'_{LL}| + |\Theta'_{UR} - \Theta'_{LR}| + |R'_{UL} - R'_{LL}| + |R'_{UR} - R'_{LR}|\end{array}}{12} \quad (11)$$

$$SYM_{radial} = \frac{\begin{array}{l}|X'_{UL} - X'_{LR}| + |X'_{UR} - X'_{LL}| + |Y'_{UL} - Y'_{LR}| + |Y'_{UR} - Y'_{LL}| + \\ |H'_{UL} - H'_{LR}| + |H'_{UR} - H'_{LL}| + |B'_{UL} - B'_{LR}| + |B'_{UR} - B'_{LL}| + \\ |\Theta'_{UL} - \Theta'_{LR}| + |\Theta'_{UR} - \Theta'_{LL}| + |R'_{UL} - R'_{LR}| + |R'_{UR} - R'_{LL}|\end{array}}{12} \quad (12)$$

where X'_j, Y'_j, H'_j, B'_j, Θ'_j, and R'_j are, respectively, the normalised values of X_j, Y_j, H_j, B_j, Θ_j, and R_j with

$$X_j = \sum_i^{n_j} |x_{ij} - x_c| \quad j = UL, UR, LL, LR \quad (13)$$

$$Y_j = \sum_i^{n_j} |y_{ij} - y_c| \quad j = UL, UR, LL, LR \quad (14)$$

$$H_j = \sum_i^{n_j} h_{ij} \quad j = UL, UR, LL, LR \quad (15)$$

$$B_j = \sum_i^{n_j} b_{ij} \quad j = UL, UR, LL, LR \quad (16)$$

$$\Theta_j = \sum_i^{n_j} \left| \frac{y_{ij} - y_c}{x_{ij} - x_c} \right| \quad j = UL, UR, LL, LR \tag{17}$$

$$R_j = \sum_i^{n_j} \sqrt{(x_{ij} - x_c)^2 + (y_{ij} - y_c)^2} \quad j = UL, UR, LL, LR \tag{18}$$

where *UL*, *UR*, *LL*, and *LR* stand for upper-left, upper-right, lower-left, and lower-right, respectively; (x_{ij},y_{ij}) and (x_c,y_c) are the co-ordinates of the centres of object i on quadrant j and the frame; b_{ij} and h_{ij} are the width and height of the object; and n_j is the total number of objects on the quadrant.

C.4 Measure of Sequence

Sequence refers to the arrangement of objects in a layout in a way that facilitates the movement of the eye through the information displayed. Sequence, by definition, is a measure of how information in a display is ordered in relation to a reading pattern that is common in Western cultures and is given by

$$SQM = 1 - \frac{\sum_{k=A,C,S} \sum_{j=UL,UR,LL,LR} \left| q_j - v_j^k \right|}{24} \quad \in [0,1] \tag{19}$$

with

$$\{q_{UL}, q_{UR}, q_{LL}, q_{LR}\} = \{4,3,2,1\} \tag{20}$$

$$v_j^k = \begin{cases} 4 & \text{if } w_j^k \text{ is the largest in } w^k \\ 3 & \text{if } w_j^k \text{ is the 2nd largest in } w^k \\ 2 & \text{if } w_j^k \text{ is the 3rd largest in } w^k \\ 1 & \text{if } w_j^k \text{ is the smallest in } w^k \end{cases} \qquad j = UL, UR, LL, LR, \quad k = A, C, S \tag{21}$$

with

$$w_j^A = q_j \sum_i^{n_j} a_{ij} \quad j = UL, UR, LL, LR \tag{22}$$

$$w_j^C = q_j \sum_i^{n_j} \left| c_{ij} - c_{frame} \right| \quad j = UL, UR, LL, LR \tag{23}$$

$$w_j^S = q_j \sum_i^{n_j} s_{ij} \quad j = UL, UR, LL, LR \tag{24}$$

$$w^k = \{w_{UL}^k, w_{UR}^k, w_{LL}^k, w_{LR}^k\} \tag{25}$$

where *UL*, *UR*, *LL*, and *LR* stand for upper-left, upper-right, lower-left, and lower-right, respectively; *A*, *C*, and *S* stand for size, colour, and shape, respectively; and a_{ij}, c_{ij}, and s_{ij} are, respectively, the area, colour, and shape of object i on quadrant j. The colour range is between 0 (white) and 1 (black). s_{ij} is calculated as per Birkhoff method. Each quadrant is given a weighting in q.

C.5 Measure of Cohesion

Similar aspect ratios promote cohesion. The term aspect ratio refers to the relationship of width to height. Typical paper sizes are higher than they are wide, while the opposite is true for typical VDU displays. Cohesion is given by

$$CM = \frac{\left| CM_{fit} \right| + \left| CM_{lo} \right|}{2} \quad \in [0,1] \tag{26}$$

CM_{fl} is a relative measure of the ratios of the layout and screen with

$$CM_{fl} = \begin{cases} t_{fl} & if\ t_{fl} \le 1 \\ \dfrac{1}{t_{fl}} & Otherwise \end{cases} \quad (27)$$

and CM_{lo} is a relative measure of the ratios of the objects and layout with

$$CM_{lo} = \frac{\sum_{i}^{n} f_i}{n} \quad (28)$$

with

$$t_{fl} = \frac{h_{layout} / b_{layout}}{h_{frame} / b_{frame}} \quad (29)$$

$$f_i = \begin{cases} t_i & if\ t_i \le 1 \\ \dfrac{1}{t_i} & Otherwise \end{cases} \quad (30)$$

with

$$t_i = \frac{h_i / b_i}{h_{layout} / b_{layout}} \quad (31)$$

where b_i and h_i, b_{layout} and h_{layout}, and b_{frame} and h_{frame} are the widths and heights of object i, the layout, and the frame, respectively; and n is the number of objects on the frame.

C.6 Measure of Unity

Unity is coherence, a totality of elements that is visually all one piece. Unity, by definition, is the extent to which the screen elements seem to belong together and is given by

$$UM = \frac{|UM_{form}| + |UM_{space}|}{2} \in [0,1] \quad (32)$$

UM_{form} is the extent to which the objects are related in size, shape, and colour with

$$UM_{form} = 1 - \frac{n_{size} + n_{colour} + n_{shape} - 3}{3n} \quad (33)$$

and UM_{space} is a relative measure of the space between groups and that of the margins with

$$UM_{space} = 1 - \frac{a_{layout} - \sum_{i}^{n} a_i}{a_{frame} - \sum_{i}^{n} a_i} \quad (34)$$

where a_i, a_{layout}, and a_{frame} are the areas of object i, the layout, and the frame, respectively; n_{size}, n_{colour}, and n_{shape} are the numbers of sizes, colours, and shapes used, respectively; and n is the number of objects on the frame.

C.7 Measure of Proportion

Down through the ages, people and cultures have had preferred proportional relationships. What constitutes beauty in one culture is not necessarily considered the same by another culture, but some proportional shapes have stood the test of time and are found in abundance today. The following shapes have been described as aesthetically pleasing.

- Square (1:1)
- Square root of two (1:1.414)
- Golden rectangle (1:1.618)
- Square root of three (1:1.732)
- Double square (1:2)

In screen design, aesthetically pleasing proportions should be considered for major components of the screen, including windows and groups of data and text.

Proportion, by definition, is the comparative relationship between the dimensions of the screen components and proportional shapes and is given by

$$PM = \frac{\left|PM_{object}\right| + \left|PM_{layout}\right|}{2} \quad \in [0,1] \tag{35}$$

PM_{object} is the difference between the proportions of the objects and the closest proportional shapes described by Marcus with

$$PM_{object} = \frac{1}{n}\sum_{i}^{n}\left(1 - \frac{\min\left(\left|p_j - p_i\right|, j = sq, r2, gr, r3, ds\right)}{0.5}\right) \tag{36}$$

and PM_{layout} is the difference between the proportions of the layout and the closest proportional shape with

$$PM_{layout} = 1 - \frac{\min\left(\left|p_j - p_{layout}\right|, j = sq, r2, gr, r3, ds\right)}{0.5} \tag{37}$$

with

$$\left\{p_{sq}, p_{r2}, p_{gr}, p_{r3}, p_{ds}\right\} = \left\{\frac{1}{1}, \frac{1}{1.414}, \frac{1}{1.618}, \frac{1}{1.732}, \frac{1}{2}\right\} \tag{38}$$

$$p_i = \begin{cases} r_i & if\ r_i \le 1 \\ \dfrac{1}{r_i} & Otherwise \end{cases} \tag{39}$$

$$p_{layout} = \begin{cases} r_{layout} & if\ r \le 1 \\ \dfrac{1}{r_{layout}} & Otherwise \end{cases} \tag{40}$$

with

$$r_i = \frac{h_i}{b_i} \tag{41}$$

$$r_{layout} = \frac{h_{layout}}{b_{layout}} \tag{42}$$

where b_i and h_i are the width and height of object i; b_{layout} and h_{layout} are the width and height of the layout; and p_j is the proportion of shape j. (Note that the maximum value of $(p_j\text{-}r_{layout})$ is 0.5.)

C.8 Measure of Simplicity

Simplicity is directness and singleness of form, a combination of elements that results in ease in comprehending the meaning of a pattern. It can be calculated as

$$SMM = \frac{3}{n_{vap} + n_{hap} + n} \quad \in [0,1] \tag{43}$$

where n_{vap} and n_{hap} are the numbers of vertical and horizontal alignment points; and n is the number of objects on the frame.

C.9 Measure of Density

Density is the extent to which the screen is covered with objects. Density is achieved by minimising screen density levels. A measure of density is the percentage of the entire frame is devoted to objects.

$$DM = 1 - \frac{\sum_{i}^{n} a_i}{a_{frame}} \quad \in [0,1] \tag{44}$$

where a_i and a_{frame} are the areas of object i and the frame; and n is the number of objects on the frame.

C.10 Measure of Regularity

Regularity is a uniformity of elements based on some principle or plan. Regularity can be calculated as

$$RM = \frac{\left|RM_{alignment}\right| + \left|RM_{spacing}\right|}{2} \quad \in [0,1] \tag{45}$$

$RM_{alignment}$ is the extent to which the alignment points are minimised with

$$RM_{alignment} = 1 - \frac{n_{vap} + n_{hap}}{2n} \tag{46}$$

and $RM_{spacing}$ is the extent to which the alignment points are consistently spaced with

$$RM_{spacing} = \begin{cases} 1 & if\ n = 1 \\ 1 - \dfrac{n_{spacing} - 1}{2(n-1)} & Otherwise \end{cases} \tag{47}$$

where n_{vap} and n_{hap} are the numbers of vertical and horizontal alignment points; $n_{spacing}$ is the number of distinct distances between column and row starting points; and n is the number of objects on the frame.

C.11 Measure of Economy

Economy is the careful and discreet use of display elements to get the message across as simple as possible. It can be calculated as

$$ECM = \frac{3}{n_{size} + n_{colour} + n_{shape}} \quad \in [0,1] \tag{48}$$

where n_{size}, n_{colour}, and n_{shape} are the numbers of sizes, colours, and shapes used, respectively.

C.12 Measure of Homogeneity

We interpret the statistical entropy concept for screen design. The entropy equation is given by the following

$$S = k \log W \tag{49}$$

where S is the entropy of the screen, k is a constant, known as Boltzmann's constant, and W is a measure of the degree of homogeneity. Homogeneity can be calculated as

$$HM = \frac{W}{W_{max}} \quad \in [0,1] \tag{50}$$

W is the number of different ways a group of n objects can be arranged for the four quadrants when n_j is the total number of objects in quadrant j, that is

$$W = \frac{n!}{\prod_{j=UL,UR,LL,LR} n_j} = \frac{n!}{n_{UL}!\,n_{UR}!\,n_{LL}!\,n_{LR}!} \tag{51}$$

W is maximum when the n objects are evenly allocated to the various quadrants of the screen, as compared to more or less uneven allocations among the quadrants, and thus

$$W_{max} = \frac{n!}{\frac{n}{4}!\frac{n}{4}!\frac{n}{4}!\frac{n}{4}!} = \frac{n!}{\left(\frac{n}{4}!\right)^4} \tag{52}$$

where n_{UL}, n_{UR}, n_{LL}, and n_{LR} are the numbers of objects on the upper-left, upper-right, lower-left, and lower-right quadrants, respectively; and n is the number of objects on the frame.

C.13 Measure of Rhythm

Rhythm in design refers to regular patterns of changes in the elements. Rhythm, by definition, is the extent to which the objects are systematically ordered and is given by

$$RHM = 1 - \frac{|RHM_x| + |RHM_y| + |RHM_{area}|}{3} \quad \in [0,1] \tag{53}$$

The rhythm components are

$$RHM_x = \frac{\begin{array}{l}|X'_{UL} - X'_{UR}| + |X'_{UL} - X'_{LR}| + \\ |X'_{UL} - X'_{LL}| + |X'_{UR} - X'_{LR}| + \\ |X'_{UR} - X'_{LL}| + |X'_{LR} - X'_{LL}|\end{array}}{6} \tag{54}$$

$$RHM_y = \frac{\begin{array}{l}|Y'_{UL} - Y'_{UR}| + |Y'_{UL} - Y'_{LR}| + \\ |Y'_{UL} - Y'_{LL}| + |Y'_{UR} - Y'_{LR}| + \\ |Y'_{UR} - Y'_{LL}| + |Y'_{LR} - Y'_{LL}|\end{array}}{6} \tag{55}$$

$$RHM_{area} = \frac{\begin{array}{l}|A'_{UL} - A'_{UR}| + |A'_{UL} - A'_{LR}| + \\ |A'_{UL} - A'_{LL}| + |A'_{UR} - A'_{LR}| + \\ |A'_{UR} - A'_{LL}| + |A'_{LR} - A'_{LL}|\end{array}}{6} \tag{56}$$

where X'_j, Y'_j, and A'_j are, respectively, the normalised values of X_j, Y_j, and A_j with

$$X_j = \sum_i^{n_j} |x_{ij} - x_c| \quad j = UL, UR, LL, LR \tag{57}$$

$$Y_j = \sum_i^{n_j} |y_{ij} - y_c| \quad j = UL, UR, LL, LR \tag{58}$$

$$A_j = \sum_i^{n_j} a_{ij} \quad j = UL, UR, LL, LR \tag{59}$$

where *UL*, *UR*, *LL*, and *LR* stand for upper-left, upper-right, lower-left, and lower-right, respectively; (x_{ij},y_{ij}) and (x_c,y_c) are the co-ordinates of the centres of object i on quadrant j and the frame; a_{ij} is the area of the object; and n_j is the total number of objects on the quadrant.

C.14 Measure of Order and Complexity

The measure of order is written as the sum of the above measures for a layout. The opposite pole on the continuum is complexity. The scale created may also be considered a scale of complexity, with extreme complexity at one end and minimal complexity (order) at the other.

The linear summation of the weighted measures designated by *OM* is given by

$$OM = \frac{\sum_i^{13} \alpha_i M_i}{13} \quad \in [0,1], \quad 0 \le \alpha_i \le 1 \tag{60}$$

with

$$M = \{M_1, M_2, M_3, M_4, M_5, M_6, M_7, M_8, M_9, M_{10}, M_{11}, M_{12}, M_{13}\}$$
$$= \{BM, EM, SYM, SQM, CM, UM, PM, SMM, DM, RM, ECM, HM, RHM\} \tag{61}$$

where BM is given by (1), EM by (6), SYM by (9), SQM by (19), CM by (26), UM by (32), PM by (35), SMM by (43), DM by (44), RM by (45), ECM by (48), HM by (50) and RHM by (53). The weighing component α is assumed to be a constant. Aesthetic measure Mi has its own weighing component αi. Choosing weights is an unresolved issue.

A special MS-Excel template was developed to automate the calculation of the measures for the models. The following tables give an overview of the Excel template formulas used to calculate the various element and diagram attributes in order to derive the aesthetics measures. The Element and Diagram Attribute names generally correspond with the formulas as given above.

C.15 Diagram Element Attribute Calculations

The diagram element attributes were calculated horizontally on the worksheet "Elements" i.e. one screen element per row (allowing for 65535 elements). Row 3 = the first screen element for the AKMA model.

Row 1: Column nr	Row 2: Element Attribute Names	Row 3 etc.: Data Sample value or formula (for row 3)
A	Mod	AK
B	Ent#	9
C	Shape	1
D	HalfHor	R
E	HalfVert	B
F	Quadrant	RB
G	ModHalfHor	AKR
H	ModHalfVert	AKB
I	ModQuadrant	AKRB
J	Gray	0.5
K	Width	31
L	Height	19
M	DistCentHor	75
N	DistCentVert	48
O	DistFrameHor	15
P	DistFrameVert	10
Q	Area	=CHOOSE(C3,1,PI()/4,0.5,0.5)*K3*L3
R	Birkhoff	=CHOOSE(C3,0.18,0,0.16,0.057143)
S	WL	0
T	WR	25.2
U	WT	0
V	WB	16.8
W	Mx	=IF(D3="R",1,-1)*M3*Q3
X	My	=IF(E3="B",1,-1)*N3*Q3
Y	F	=IF(M3=0,PI()/2,ATAN(ABS(N3/M3)))
Z	R	=SQRT(M3^2+N3^2)
AA	ti	1.051461988
AB	fi	0.95105673
AC	ri	0.612903226
AD	Pmobject	=MIN(ABS(AC3-1),ABS(AC3-SQRT(0.5)),ABS(AC3-1/1.618),ABS(AC3-SQRT(1/3)),ABS(AC3-0.5))

C.16 Diagram Attribute Calculations

These were calculated on the worksheet "Models" in vertical format: one column for each model allowing for 254 models. (column C=AK=AKMA)

Col A: Row Nr	Col B: Diagram Attribute Name	Col C and following: Data Sample value or Excel Formula (for column C)
1	Column Number	2
2		Mod
3	**Mod**	**AK**
4	Diagram Width	180
5	Diagram Height	116
6	Diagram Area	=C4*C5
7	Diagram W/H Ratio	=C4/C5
8	Sum Elements Area	=DSUM(Elements!$A$2:$Q$117,"Area",Models!C$2:C$3)
9	Nr elements	=COUNTIF(Elements!$A$3:$A$117,Models!C3)
10	***Balance***	

11		ModHalfHor
12	L	=C$3&$B12
13	a(Max,L)	=DMAX(Elements!$A$2:$R$117,17,Models!C11:C12)
14		ModHalfHor
15	R	=C$3&$B15
16	a(Max,R)	=DMAX(Elements!$A$2:$R$117,17,Models!C14:C15)
17		ModHalfVert
18	T	=C$3&$B18
19	a(Max,T)	=DMAX(Elements!$A$2:$R$117,17,Models!C17:C18)
20		ModHalfVert
21	B	=C$3&$B21
22	a(Max,B)	=DMAX(Elements!$A$2:$R$117,17,Models!C20:C21)
23	WL	=DSUM(Elements!$A$2:$V$117,$B23,Models!C$2:C$3)
24	WR	=DSUM(Elements!$A$2:$V$117,$B24,Models!C$2:C$3)
25	WT	=DSUM(Elements!$A$2:$V$117,$B25,Models!C$2:C$3)
26	WB	=DSUM(Elements!$A$2:$V$117,$B26,Models!C$2:C$3)
27	BMHorizontal	=ABS(C23-C24)/MAX(C23,C24)
28	BMVertical	=ABS(C25-C26)/MAX(C25,C26)
29	BM	=1-(C27+C28)/2
30	***Equilibrum***	
31	Emx	=2*DSUM(Elements!$A$2:$X$117,"Mx",Models!C$2:C$3)/C$9/C4/C8
32	Emy	=2*DSUM(Elements!$A$2:$X$117,"My",Models!C$2:C$3)/C$9/C5/C8
33	EM	=1-(ABS(C31)+ABS(C32))/2
34	***Symmetry***	
35		ModQuadrant
36	LB	=C$3&$B36
37		ModQuadrant
38	RB	=C$3&$B38
39		ModQuadrant
40	LT	=C$3&$B40
41		ModQuadrant
42	RT	=C$3&$B42
43	BBL	=DAVERAGE(Elements!$A$2:$Z$117,"Width",C35:C36)
44	BBLn	=C43/C$4/2
45	BBR	=DAVERAGE(Elements!$A$2:$Z$117,"Width",C37:C38)
46	BBRn	=C45/C$4/2
47	BTL	=DAVERAGE(Elements!$A$2:$Z$117,"Width",C39:C40)
48	BTLn	=C47/C$4/2
49	BTR	=DAVERAGE(Elements!$A$2:$Z$117,"Width",C41:C42)
50	BTRn	=C49/C$4/2
51	FBL	=DAVERAGE(Elements!$A$2:$Z$117,"F",C35:C36)
52	FBLn	=C51*2/PI()
53	FBR	=DAVERAGE(Elements!$A$2:$Z$117,"F",C37:C38)
54	FBRn	=C53*2/PI()
55	FTL	=DAVERAGE(Elements!$A$2:$Z$117,"F",C39:C40)
56	FTLn	=C55*2/PI()
57	FTR	=DAVERAGE(Elements!$A$2:$Z$117,"F",C41:C42)
58	FTRn	=C57*2/PI()
59	HBL	=DAVERAGE(Elements!$A$2:$Z$117,"Height",C35:C36)
60	HBLn	=C59/C$5/2
61	HBR	=DAVERAGE(Elements!$A$2:$Z$117,"Height",C37:C38)
62	HBRn	=C61/C$5/2

63	HTL	=DAVERAGE(Elements!$A$2:$Z$117,"Height",C39:C40)
64	HTLn	=C63/C$5/2
65	HTR	=DAVERAGE(Elements!$A$2:$Z$117,"Height",C41:C42)
66	HTRn	=C65/C$5/2
67	RBL	=DAVERAGE(Elements!$A$2:$Z$117,"R",C35:C36)
68	RBLn	=C67*2/SQRT(C4^2+C5^2)
69	RBR	=DAVERAGE(Elements!$A$2:$Z$117,"R",C37:C38)
70	RBRn	=C69*2/SQRT(C4^2+C5^2)
71	RTL	=DAVERAGE(Elements!$A$2:$Z$117,"R",C39:C40)
72	RTLn	=C71*2/SQRT(C4^2+C5^2)
73	RTR	=DAVERAGE(Elements!$A$2:$Z$117,"R",C41:C42)
74	RTRn	=C73*2/SQRT(C4^2+C5^2)
75	XLL	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentHor",C35:C36)
76	XLLn	=C75/C$4/2
77	XLR	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentHor",C37:C38)
78	XLRn	=C77/C$4/2
79	XUL	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentHor",C39:C40)
80	XULn	=C79/C$4/2
81	XUR	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentHor",C41:C42)
82	XURn	=C81/C$4/2
83	YBL	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentVert",C35:C36)
84	YBLn	=C83/C$5/2
85	YBR	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentVert",C37:C38)
86	YBRn	=C85/C$5/2
87	YTL	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentVert",C39:C40)
88	YTLn	=C87/C$5/2
89	YTR	=DAVERAGE(Elements!$A$2:$Z$117,"DistCentVert",C41:C42)
90	YTRn	=C89/C$5/2
91	SYMvertical	=(ABS(C80-C82)+ABS(C76-C78)+ABS(C88-C90)+ABS(C84-C86)+ABS(C64-C66)+ABS(C60-C62)+ABS(C48-C50)+ABS(C44-C46)+ABS(C56-C58)+ABS(C52-C54)+ABS(C72-C74)+ABS(C68-C70))/12
92	SYMhorizontal	=(ABS(C84-C88)+ABS(C86-C90)+ABS(C76-C80)+ABS(C78-C82)+ABS(C68-C72)+ABS(C70-C74)+ABS(C60-C64)+ABS(C62-C66)+ABS(C52-C56)+ABS(C54-C58)+ABS(C44-C48)+ABS(C46-C50))/12
93	SYMradial	=(ABS(C84-C90)+ABS(C86-C900)+ABS(C76-C82)+ABS(C78-C80)+ABS(C68-C74)+ABS(C70-C72)+ABS(C60-C66)+ABS(C62-C64)+ABS(C44-C50)+ABS(C46-C48)+ABS(C52-C58)+ABS(C54-C56))/12
94	SYM	=1-SUM(C91:C93)/3
95	***Sequence***	
96	W A , LB	=DSUM(Elements!$A$2:$Z$117,"Area",C$35:C$36)*2
97	W A , RB	=DSUM(Elements!$A$2:$Z$117,"Area",C$37:C$38)
98	W A , LT	=DSUM(Elements!$A$2:$Z$117,"Area",C$39:C$40)*4
99	W A , RT	=DSUM(Elements!$A$2:$Z$117,"Area",C$41:C$42)*3
100	W C , LB	=DSUM(Elements!$A$2:$Z$117,"Gray",C$35:C$36)*2
101	W C , RB	=DSUM(Elements!$A$2:$Z$117,"Gray",C$37:C$38)
102	W C , LT	=DSUM(Elements!$A$2:$Z$117,"Gray",C$39:C$40)*4
103	W C , RT	=DSUM(Elements!$A$2:$Z$117,"Gray",C$41:C$42)*3
104	W S , LB	=DSUM(Elements!$A$2:$Z$117,"Shape",C$35:C$36)*2
105	W S , RB	=DSUM(Elements!$A$2:$Z$117,"Shape",C$37:C$38)
106	W S , LT	=DSUM(Elements!$A$2:$Z$117,"Shape",C$39:C$40)*4
107	W S , RT	=DSUM(Elements!$A$2:$Z$117,"Shape",C$41:C$42)*3
108	1	=LARGE(C$96:C$99,$B108)

109	2	=LARGE(C$96:C$99,$B109)
110	3	=LARGE(C$96:C$99,$B110)
111	4	=LARGE(C$96:C$99,$B111)
112	1	=LARGE(C$100:C$103,$B112)
113	2	=LARGE(C$100:C$103,$B113)
114	3	=LARGE(C$100:C$103,$B114)
115	4	=LARGE(C$100:C$103,$B115)
116	1	=LARGE(C$104:C$107,$B116)
117	2	=LARGE(C$104:C$107,$B117)
118	3	=LARGE(C$104:C$107,$B118)
119	4	=LARGE(C$104:C$107,$B119)
120		11
121	V A , LB	=VLOOKUP(C96,C$108:$M$111,C$120,FALSE)
122	V A , RB	=VLOOKUP(C97,C$108:$M$111,C$120,FALSE)
123	V A , LT	=VLOOKUP(C98,C$108:$M$111,C$120,FALSE)
124	V A , RT	=VLOOKUP(C99,C$108:$M$111,C$120,FALSE)
125	V C , LB	=VLOOKUP(C100,C$112:$M$115,C$120,FALSE)
126	V C , RB	=VLOOKUP(C101,C$112:$M$115,C$120,FALSE)
127	V C , LT	=VLOOKUP(C102,C$112:$M$115,C$120,FALSE)
128	V C , RT	=VLOOKUP(C103,C$112:$M$115,C$120,FALSE)
129	V S , LB	=VLOOKUP(C104,C$116:$M$119,C$120,FALSE)
130	V S , RB	=VLOOKUP(C105,C$116:$M$119,C$120,FALSE)
131	V S , LT	=VLOOKUP(C106,C$116:$M$119,C$120,FALSE)
132	V S , RT	=VLOOKUP(C107,C$116:$M$119,C$120,FALSE)
133	Diff A , LB	=ABS(C121-$M133)
134	Diff A , RB	=ABS(C122-$M134)
135	Diff A , LT	=ABS(C123-$M135)
136	Diff A , RT	=ABS(C124-$M136)
137	Diff C , LB	=ABS(C125-$M137)
138	Diff C , RB	=ABS(C126-$M138)
139	Diff C , LT	=ABS(C127-$M139)
140	Diff C , RT	=ABS(C128-$M140)
141	Diff S , LB	=ABS(C129-$M141)
142	Diff S , RB	=ABS(C130-$M142)
143	Diff S , LT	=ABS(C131-$M143)
144	Diff S , RT	=ABS(C132-$M144)
145	SQM	=SUM(C133:C144)/24
146	***Cohesion***	
147	Width Frame	
148	Height Frame	
149	Frame W/H Ratio	=4/3
150	Diagram Ratio	=C7
151	CMfl	=IF(C150>C149,C149/C150,C150/C149)
152	CMlo	=DAVERAGE(Elements!$A$2:$AB$117,"fi",C2:C3)
153	CM	=(C151+C152)/2
154	***Unity***	
155	N size	9
156	N colour	1
157	N shape	1
158	UM form	=1-(SUM(C155:C157)-3)/3/C9
159	Layout Area - Sum El Area	=C6-C8
160	Frame Area	=IF(C149=4/3,IF(C149<C150,C4*C4/C149,C5*C5*C149),C147*C148)

161	Frame Area - Sum El Area	=C160-C8
162	UM space	=1-(C159/C161)
163	UM	=(C162+C158)/2
164	***Proportion***	
165	p layout	=IF(C150>1,1/C150,C150)
166	PM layout	=MIN(ABS(C165-1),ABS(C165-SQRT(0.5)),ABS(C165-1/1.618),ABS(C165-SQRT(1/3)),ABS(C165-0.5))
167	PM object	=DAVERAGE(Elements!$A$2:$AD$117,"Pmobject",C2:C3)
168	PM	=(C166+C167)/2
169	***Simplicity***	
170	n joint hap	1
171	n joint vap	4
172	n hap	=C$9-C170
173	n vap	=C$9-C171
174	n	=C9
175	SMM	=3/SUM(C172:C174)
176	***Density***	
177	DM	=1-C8/C6
178	***Regularity***	
179	RM alignment	=1-(C172+C173)/C174/2
180	n spacing	9
181	RM spacing	=IF(C180=1,1,1-(C180-1)/2/(C174-1))
182	RM	=(ABS(C179)+ABS(C181))/2
183	***Economy***	
184	ECM	=3/SUM(C155:C157)
185	***Homogeneity***	
186	N LB	=COUNTIF(Elements!$I$3:$I$117,Models!C36)
187	N LB !	=FACT(C186)
188	N RB	=COUNTIF(Elements!$I$3:$I$117,Models!C38)
189	N RB !	=FACT(C188)
190	N LT	=COUNTIF(Elements!$I$3:$I$117,Models!C40)
191	N LT !	=FACT(C190)
192	N RT	=COUNTIF(Elements!$I$3:$I$117,Models!C42)
193	N RT !	=FACT(C192)
194	W	=FACT(C9)/C187/C189/C191/C193
195	W max	=FACT(C9)/(FACT(C9/4)^4)
196	HM	=C194/C195
197	***Rhythm***	
198	RHM x	=(ABS(C76-C78)+ABS(C76-C80)+ABS(C76-C82)+ABS(C78-C80)+ABS(C78-C82)+ABS(C80-C82))/6
199	RHM y	=(ABS(C84-C86)+ABS(C84-C88)+ABS(C84-C90)+ABS(C86-C88)+ABS(C86-C90)+ABS(C88-C90))/6
200	A LB	=DSUM(Elements!$A$2:$Z$117,"Area",C35:C36)
201	A LBn	=C200/C$6/4
202	A RB	=DSUM(Elements!$A$2:$Z$117,"Area",C37:C38)
203	A RBn	=C202/C$6/4
204	A LT	=DSUM(Elements!$A$2:$Z$117,"Area",C39:C40)
205	A LTn	=C204/C$6/4
206	A RT	=DSUM(Elements!$A$2:$Z$117,"Area",C41:C42)
207	A RTn	=C206/C$6/4
208	A total	=C200+C202+C204+C206
209	RHM area	=(ABS(C201-C203)+ABS(C201-C205)+ABS(C201-C207)+ABS(C203-C205)+ABS(C203-C207)+ABS(C205-C207))/6

210	RHM	=1-(C209+C199+C198)/3
211	***Order & Complexity***	
212	Balance	=VLOOKUP($B212,$B$1:$M$210,C$1,FALSE)
213	Equilibrum	=VLOOKUP($B213,$B$1:$M$210,C$1,FALSE)
214	Symmetry	=VLOOKUP($B214,$B$1:$M$210,C$1,FALSE)
215	Sequence	=VLOOKUP($B215,$B$1:$M$210,C$1,FALSE)
216	Cohesion	=VLOOKUP($B216,$B$1:$M$210,C$1,FALSE)
217	Unity	=VLOOKUP($B217,$B$1:$M$210,C$1,FALSE)
218	Proportion	=VLOOKUP($B218,$B$1:$M$210,C$1,FALSE)
219	Simplicity	=VLOOKUP($B219,$B$1:$M$210,C$1,FALSE)
220	Density	=VLOOKUP($B220,$B$1:$M$210,C$1,FALSE)
221	Regularity	=VLOOKUP($B221,$B$1:$M$210,C$1,FALSE)
222	Homogeneity	=VLOOKUP($B222,$B$1:$M$210,C$1,FALSE)
223	Economy	=VLOOKUP($B223,$B$1:$M$210,C$1,FALSE)
224	Rhythm	=VLOOKUP($B224,$B$1:$M$210,C$1,FALSE)
225	Order/Complexity	**=SUM(C212:C224)/13**

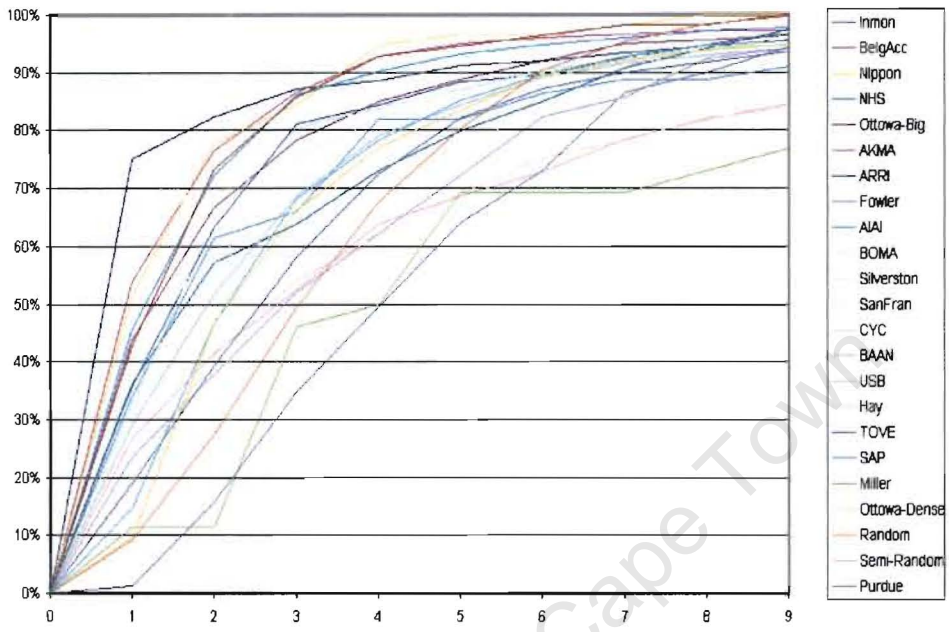
C.17 Constants for the various model element shapes.

	M	O	C	s
Rectangle	1.5	6	4	0.1800
Circle	0	4	10000000	0.0000
Diamond	1.25	5	4	0.1600
Triangle	0.166667	1	6	0.057143

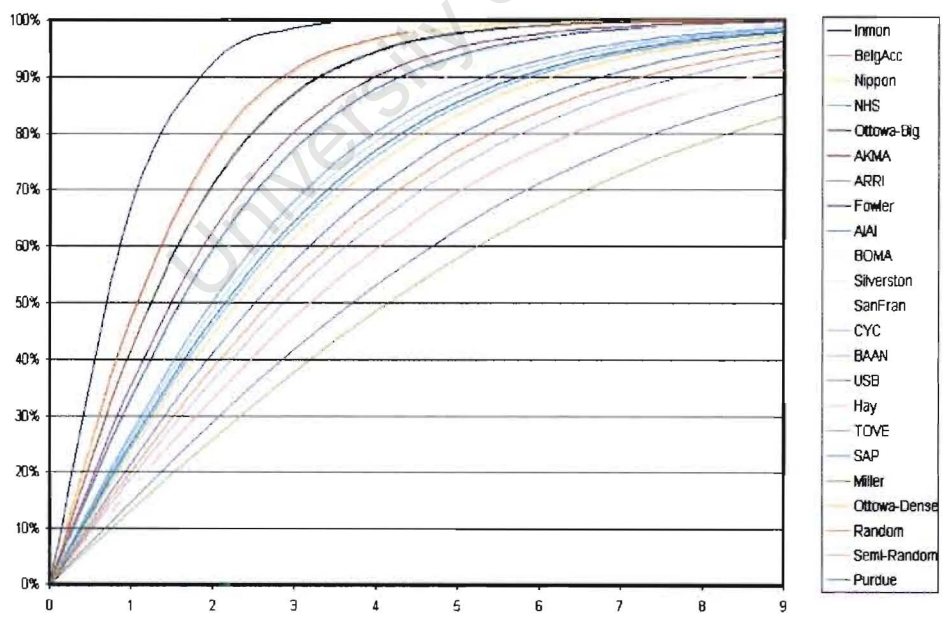
APPENDIX D: BEST FIT FOR THE CUMULATIVE FANOUT DISTRIBUTIONS

The following two charts plot the actual cumulative fan-out frequencies as well as the fitted curve using the TANH(x/h) function, for all the models in the database. A subset of some typical models is given in figure 7-5 in the thesis.

Cumulative Fanout Frequencies



Best Fit for Cumulative Fanout Frequencies Using the *y = TANH(x/h)* function



APPENDIX E: RANK CORRELATIONS BETWEEN MODEL RANKINGS FOR DIFFERENT ULS AND PCS

E.1 Rank-correlation for the different models when changing UL.

Model	BL -> MW	BL -> OB	BL -> SB	BL -> WP	BL -> All	MW -> OB	MW -> SB	MW -> WP	OB -> SB	OB -> WP	SB -> WP
AI	9	1	0	1	4	4	9	4	1	0	1
AK	64	81	100	81	1	1	4	1	1	0	1
AR	1	9	1	4	9	4	4	1	16	1	9
BA	64	16	49	16	16	16	1	16	9	0	9
BE	0	0	0	0	0	0	0	0	0	0	0
BO	25	36	9	36	49	1	4	1	9	0	9
CY	25	25	25	36	16	0	0	1	0	1	1
FO	16	16	25	4	4	0	1	36	1	36	49
HA	1	16	36	1	25	9	25	0	4	9	25
IN	1	9	16	4	4	16	25	9	1	1	4
MI	0	0	0	0	0	0	0	0	0	0	0
NH	16	4	25	4	9	4	1	4	9	0	9
NI	25	4	9	36	9	9	64	1	25	16	81
OB	225	324	225	225	324	9	0	0	9	9	0
OD	256	225	256	225	256	1	0	1	1	0	1
PU	9	36	4	49	16	9	1	16	16	1	25
RA	0	0	0	0	0	0	0	0	0	0	0
SA	100	49	49	100	49	9	9	0	0	9	9
SF	25	16	1	25	9	1	16	0	9	1	16
SI	0	9	49	16	1	9	49	16	16	1	9
SR	289	256	256	256	289	1	1	1	0	0	0
TO	0	0	1	0	0	0	1	0	1	0	1
US	0	1	0	4	1	1	0	4	1	9	4
Sum	1151	1133	1136	1123	1091	104	215	112	129	94	263
r' =	0.43	0.44	0.44	0.45	0.461	0.95	0.89	0.94	0.94	0.95	0.87
z =	2.02	2.06	2.06	2.09	2.16	4.45	4.19	4.43	4.39	4.47	4.08

This table details the relative contribution of each model to changes the overall (in all cases highly significant) rank-correlation coefficients, as explained in the perspicuity analysis section of chapter 8.
The table should be read as per following example:
When using the MW lexicon instead of the BL lexicon (first data column), the relative ranking of say the AIAI model (first data row) changed from 14th to 11th position, i.e. a relative change of 3 positions. Its contribution to the sum is therefore $3^2 = 9$, which is a fairly low value i.e. would not necessarily be expected if there was no (or little) correlation between the two rankings.. Note the very large contribution of the Ottawa-derived models OB, OD and SR which reduce the rank-correlation coefficient and thus the significance.
The rank-correlation is calculated as

$$r' = 1 - (6 * \text{sum of squared rank changes}) / [n * (n^2 - 1)] \quad \text{with } n = 23$$

and the z-score as follows:

$$z = r' \ / \ \text{square root } (n-1)$$

E.2 Correlation between the different PC formulas.

Model	Rankings for different PCs					Rank-differences squared				
	GPC rank	WPC rank	RAWPC rank	NRAWPC rank	RAWPC rank (WL-based)	GPC->WPC	WPC->RAWPC	RAWPC->NRAWPC	GPC->NRAWPC	BLRank<=>WL
AI	16	14	15	15	16	4	1	0	1	1
AK	5	6	10	12	12	1	16	4	49	4
AR	15	16	8	8	15	1	64	0	49	49
BA	2	1	2	2	4	1	1	0	0	4
BE	23	23	23	23	23	0	0	0	0	0
BO	8	8	7	7	5	0	1	0	1	4
CY	12	13	13	13	8	1	0	0	1	25
FO	13	12	16	16	10	1	16	0	9	36
HA	7	7	11	9	9	0	16	4	4	4
IN	9	9	12	10	13	0	9	4	1	1
MI	21	21	21	21	21	0	0	0	0	0
NH	14	15	14	14	14	1	1	0	0	0
NI	10	10	5	5	11	0	25	0	25	36
OB	19	19	19	17	19	0	0	4	4	0
OD	17	17	18	19	17	0	1	1	4	1
PU	6	4	4	4	6	4	0	0	4	4
RA	22	22	22	22	22	0	0	0	0	0
SA	3	2	3	1	2	1	1	4	4	1
SF	11	11	9	11	7	0	4	4	0	4
SI	1	5	1	3	3	16	16	4	4	4
SR	17	18	17	18	17	1	1	1	1	0
TO	20	20	20	20	20	0	0	0	0	0
US	4	3	6	6	1	1	9	0	4	25
					sum	33	182	30	165	203
					r' =	0.98	0.91	0.99	0.92	0.9
					z =	4.61	4.27	4.62	4.31	4.22

The above table investigates the changes in relative positions which occur when changing the formula for calculating the perspicuity count as presented in chapter 8. It can be seen that the changes in relative position (the first 5 data columns) are relatively small.
This is supported statistically by the fact that the model rankings based on the PCs as calculated by the different formulas are statistically highly significantly correlated: the rank-correlation coefficients are all between 0.91 and 0.99 (with 1.00 being perfectly correlated). The 5 data columns to the right of the table calculate the contributions to the rank-correlation coefficients i.e. the square of the extremely small changes in ranking. This results in very small sum values and thus very high rank-correlation coefficients. The corresponding z-scores are all above 4!

APPENDIX F: MODEL AND DICTIONARY ABBREVIATIONS USED IN THE TABLES

F.1 Model acronyms used

2 letter code	Short name	Full name
AI	AIAI	The Enterprise Ontology
AK	AKMA	AKMA's Generic DataFrame
AR	ARRI	ARRI's Small Integrated Manufacturing Enterprise Model
BA	BAAN	The Baan IV DEM Business Reference Model
BE	BelgAcc	The Belgium Accounting Framework
BO	BOMA	Chris Marshall's BOMA Model
CY	CYC	The Cyc Ontology
FO	Fowler	Fowler's Object-oriented Analysis Patterns
HA	Hay	Hay's Data Model Patterns
IN	Inmon	Bill Inmon's High and Mid-level Data Models
MI	Miller	J.G. Miller's General Living Systems Model
NH	NHS	NHS Generic HCM Class Model
NI	Nippon	The Japanese Industrial Automation System Model
OB	Ottowa-Big	Ottowa's Business Dictionary Hyperlinks Model
OD	Ottowa-Dense	Ottowa's Business Dictionary Denser Hyperlinks Model
PU	Purdue	The Purdue Reference Model for CIM
RA	Random	A fully random model
SA	SAP	The SAP R/3 Reference Model
SR	Semi-Random	A model with business entities and random relationships
SF	SanFran	San Francisco Application Business Components
SI	Silverston	Silverston et al's Universal Data Models
TO	TOVE	The TOVE ("Toronto Virtual Enterprise") Ontologies
US	USB	The U.S.B. Growth Model

F.2 Dictionaries used for semantic analysis

2 letter code	Full dictionary name
BL	BLC or Business Language Corpus
MW	MoneyWords
OB	Ottawa Journal Business Dictionary
SB	Dictionary of Small Business
WP	Washington Post Business Glossary

APPENDIX G: TYPICAL XMI TAGS

The following list of required XMI gives an indication of the complexity of even the simplest XMI specification. These specifications originate from the 10/25/1999 ad/99-10-02: XML Metadata Interchange 6-55 specification.

Every XMI-compliant DTD must include the declarations of the following XML elements:
- XMI
- XMI.header
- XMI.content
- XMI.extensions
- XMI.extension
- XMI.documentation
- XMI.owner
- XMI.contact
- XMI.longDescription
- XMI.shortDescription
- XMI.exporter
- XMI.exporterVersion
- XMI.exporterID
- XMI.notice
- XMI.model
- XMI.metamodel
- XMI.metametamodel
- XMI.import
- XMI.difference
- XMI.delete
- XMI.add
- XMI.replace
- XMI.reference

The following declarations are required if used by the particular metamodel:
- XMI.field
- XMI.struct
- XMI.seqItem
- XMI.sequence
- XMI.arrayLen
- XMI.array
- XMI.enum
- XMI.discrim
- XMI.union
- XMI.any

APPENDIX H: DOCUMENTATION FOR THE XML VERSION OF THE MODEL DATABASE

H.1 List of the XML tags used in the XML database and their mapping to the meta-model.

The table below lists all the XML tags used in the database and maps them to their meta-model equivalent. Refer to the text (chapter 5) for additional notes and comments.

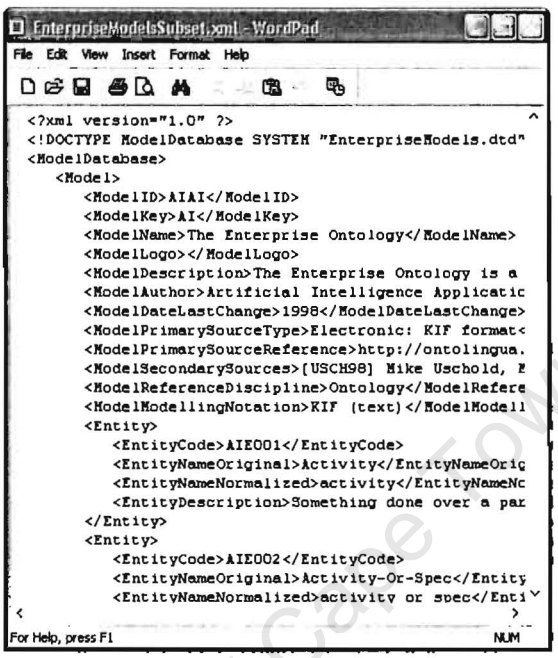
XML Tag	Meta Model Equivalent
ModelDatabase	{class containing all GenericEnterpriseModels}
Model	GenericEnterpriseModel
ModelID	GenericEnterpriseModel.Mnemonic (see appendix F)
ModelKey	{two letter abbreviation of ModelID} (see appendix F)
ModelName	GenericEnterpriseModel.Name
ModelLogo	GenericEnterpriseModel.Logo {left as null value}
ModelDescription	GenericEnterpriseModel.Description
ModelAuthor	GenericEnterpriseModel.Author/CopyrightOwner
ModelDateLastChange	GenericEnterpriseModel.DateofLastChange
ModelPrimarySourceType	GenericEnterpriseModel.PrimarySourceType
ModelPrimarySourceReference	GenericEnterpriseModel.PrimarySourceReference
ModelSecondarySources	GenericEnterpriseModel.SecondarySources
ModelReferenceDiscipline	GenericEnterpriseModel.ReferenceDiscipline
ModelModellingNotation	GenericEnterpriseModel.ModellingNotation
Entity	Entity
EntityCode	Entity.InternalCode
EntityNameOriginal	Entity.Name
EntityNameNormalized	{Entity.Name after being processed as described in chapter 8}
EntityDescription	Entity.DefinitionOrDescription
Group	GrouperOrDiagram
GroupCode	GrouperOrDiagram.InternalCode
GroupName	GrouperOrDiagram.Name
GroupDescription	{GrouperOrDiagram.DefinitionOrDescription}
IsaRel	StructuralRelationship
IsaRelCode	StructuralRelationship.InternalCode
IsaRelSubEntity	StructuralRelationship.FromEntity.InternalCode
IsaRelSuperEntity	StructuralRelationship.ToEntity.InternalCode
Relationship	DomainRelationship
RelationshipName	DomainRelationship.Name
RelationshipFromEntity	DomainRelationship.FromEntity.InternalCode
RelationshipFromEntityRole	DomainRelationship.FromRoleName
RelationshipFromCardinality	DomainRelationship.FromCardinality
RelationshipToEntity	DomainRelationship.ToEntity.InternalCode
RelationshipToEntityRole	DomainRelationship.ToRoleName
RelationshipToCardinality	DomainRelationship.ToCardinality
RelationshipType	DomainRelationship.Type
RelationshipDescription	{DomainRelationship.DefinitionOrDescription

H.2 Screenshots of the XML database using different types of editors.

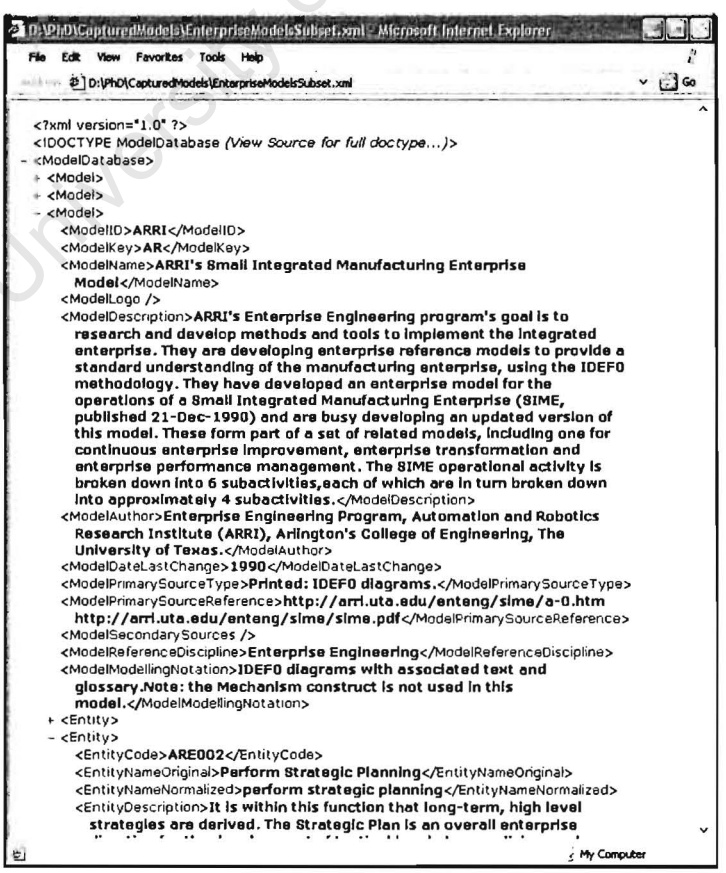
The XML Model database viewed with three different applications:

- Text editor: WordPad v 5.1 by Microsoft
- Web browser: Internet Explorer v 6.0 by Microsoft
- XML editor: XMLViewer v 1.01 by MindFusion

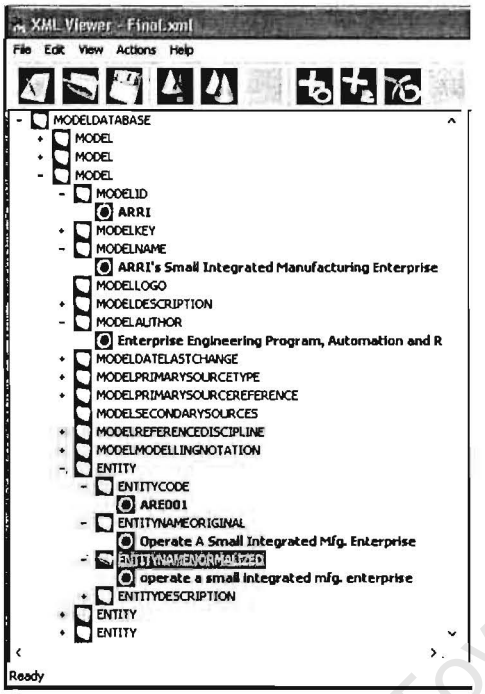
Microsoft WordPad version 5.1



Microsoft's Internet Explorer v. 6.0 (XML-enabled web browser)



MindFusion's XMLViewer v 1.01 (XML editor)



H.3 The Document Type Definition (DTD) for the XML model database.

The following is a listing of the full code of the external DTD (wordwrap applied for legibility)

```
<!--  DTD for EnterpriseModels.xml -->

<!ELEMENT ModelDatabase (Model*) >
<!ELEMENT Model (ModelID, ModelKey, ModelName, ModelLogo, ModelDescription,
ModelAuthor, ModelDateLastChange, ModelPrimarySourceType,
ModelPrimarySourceReference, ModelSecondarySources,
ModelReferenceDiscipline, ModelModellingNotation, ( Entity | Relationship |
IsaRel | Group )* ) >

<!ELEMENT ModelID  (#PCDATA)>
<!ELEMENT ModelKey  (#PCDATA)>
<!ELEMENT ModelName  (#PCDATA)>
<!ELEMENT ModelLogo  (#PCDATA)>
<!ELEMENT ModelDescription  (#PCDATA)>
<!ELEMENT ModelAuthor  (#PCDATA)>
<!ELEMENT ModelDateLastChange  (#PCDATA)>
<!ELEMENT ModelPrimarySourceType  (#PCDATA)>
<!ELEMENT ModelPrimarySourceReference  (#PCDATA)>
<!ELEMENT ModelSecondarySources  (#PCDATA)>
<!ELEMENT ModelReferenceDiscipline  (#PCDATA)>
<!ELEMENT ModelModellingNotation  (#PCDATA)>

<!ELEMENT Relationship  (RelationshipName, RelationshipFromEntity,
RelationshipFromEntityRole, RelationshipFromCardinality,
RelationshipToEntity, RelationshipToEntityRole, RelationshipToCardinality,
RelationshipType, RelationshipDescription)>
<!ELEMENT RelationshipName  (#PCDATA)>
<!ELEMENT RelationshipFromEntity  (#PCDATA)>
<!ELEMENT RelationshipFromEntityRole  (#PCDATA)>
<!ELEMENT RelationshipFromCardinality  (#PCDATA)>
<!ELEMENT RelationshipToEntity  (#PCDATA)>
<!ELEMENT RelationshipToEntityRole  (#PCDATA)>
<!ELEMENT RelationshipToCardinality  (#PCDATA)>
<!ELEMENT RelationshipType  (#PCDATA)>
```

```
<!ELEMENT RelationshipDescription  (#PCDATA)>

<!ELEMENT IsaRel  (IsaRelCode, IsaRelSubEntity, IsaRelSuperEntity)>
<!ELEMENT IsaRelCode  (#PCDATA)>
<!ELEMENT IsaRelSubEntity  (#PCDATA)>
<!ELEMENT IsaRelSuperEntity  (#PCDATA)>

<!ELEMENT Group  (GroupCode, GroupName, GroupDescription)>
<!ELEMENT GroupCode  (#PCDATA)>
<!ELEMENT GroupName  (#PCDATA)>
<!ELEMENT GroupDescription  (#PCDATA)>

<!ELEMENT Entity  (EntityCode, EntityNameOriginal, EntityNameNormalized,
EntityDescription)>
<!ELEMENT EntityCode  (#PCDATA)>
<!ELEMENT EntityNameOriginal  (#PCDATA)>
<!ELEMENT EntityNameNormalized  (#PCDATA)>
<!ELEMENT EntityDescription  (#PCDATA)>
```

Some of the code is explained below.

```
<!ELEMENT ModelDatabase (Model*) >
```

The `ModelDatabase` consists of several `Models`

```
<!ELEMENT Model (ModelID, ModelKey, ModelName, ModelLogo, ModelDescription,
ModelAuthor, ModelDateLastChange, ModelPrimarySourceType,
ModelPrimarySourceReference, ModelSecondarySources,
ModelReferenceDiscipline, ModelModellingNotation, ( Entity | Relationship |
IsaRel | Group )* ) >
```

A `Model` consists of a `ModelID, ModelKey, ... ModelModellingNotation`, and followed by any combination of `Entities, Relationships, IsaRels`, and/or `Groups`.

```
<!ELEMENT ModelID  (#PCDATA)>
```

A `ModelID` is Parsed Character Data.

```
<!ELEMENT Entity  (EntityCode, EntityNameOriginal, EntityNameNormalized,
EntityDescription)>
```

An `Entity` consists of an `EntityCode, EntityNameOriginal, EntityNameNormalized` and `EntityDescription`.

H.4 The DTD as an XML schema form

It is a trivial exercise to convert the DTD to a proper XML schema. Below is a minimal schema for the ModelDatabase, according to the 2001 schema proposal. This could be refined by defining user-defined types for e.g. EntityCodes etc.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="ModelDatabase" type="ModelDatabaseData"/>

    <xsd:complexType name="ModelDatabaseData">
       <xsd:sequence>
          <xsd:element name="Model" type="ModelData" minOccurs="1"
maxOccurs="unbounded"/>
       </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="ModelData">
         <xsd:element name="ModelID" type="xsd:string"/>
         <xsd:element name="ModelKey" type="xsd:string"/>
         <xsd:element name="ModelName" type="xsd:string"/>
         <xsd:element name="ModelLogo" type="xsd:string"/>
         <xsd:element name="ModelDescription" type="xsd:string"/>
```

```
        <xsd:element name="ModelAuthor" type="xsd:string"/>
        <xsd:element name="ModelDateLastChange" type="xsd:gYear"/>
        <xsd:element name="ModelPrimarySourceType" type="xsd:string"/>
        <xsd:element name="ModelPrimarySourceReference" type="xsd:string"/>
        <xsd:element name="ModelSecondarySources" type="xsd:string"/>
        <xsd:element name="ModelReferenceDiscipline" type="xsd:string"/>
        <xsd:element name="ModelModellingNotation" type="xsd:string"/>
      <xsd:sequence>
        <xsd:element name="Entity" type="EntityData" minOccurs="0"
maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:sequence>
        <xsd:element name="Group" type="GroupData" minOccurs="0"
maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:sequence>
        <xsd:element name="IsaRel" type="IsaRelData" minOccurs="0"
maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:sequence>
        <xsd:element name="Relationship" type="RelationshipData" minOccurs="0"
maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="EntityData">
        <xsd:element name="EntityCode" type="xsd:string"/>
        <xsd:element name="EntityNameOriginal" type="xsd:string"/>
        <xsd:element name="EntityNameNormalized" type="xsd:string"/>
        <xsd:element name="EntityDescription" type="xsd:string"/>
    </xsd:complexType>

    <xsd:complexType name="GroupData">
        <xsd:element name="GroupCode" type="xsd:string"/>
        <xsd:element name="GroupName" type="xsd:string"/>
        <xsd:element name="GroupDescription" type="xsd:string"/>
    </xsd:complexType>

    <xsd:complexType name="EntityData">
        <xsd:element name="EntityCode" type="xsd:string"/>
        <xsd:element name="EntityNameOriginal" type="xsd:string"/>
        <xsd:element name="EntityNameNormalized" type="xsd:string"/>
        <xsd:element name="EntityDescription" type="xsd:string"/>
    </xsd:complexType>

    <xsd:complexType name="IsaRelData">
        <xsd:element name="IsaRelCode" type="xsd:string"/>
        <xsd:element name="IsaRelSubEntity" type="xsd:string"/>
        <xsd:element name="IsaRelSuperEntity" type="xsd:string"/>
    </xsd:complexType>

    <xsd:complexType name="RelationshipData">
        <xsd:element name="RelationshipName" type="xsd:string"/>
        <xsd:element name="RelationshipFromEntity" type="xsd:string"/>
        <xsd:element name="RelationshipFromEntityRole" type="xsd:string"/>
        <xsd:element name="RelationshipFromCardinality" type="xsd:string"/>
        <xsd:element name="RelationshipToEntity" type="xsd:string"/>
        <xsd:element name="RelationshipToEntityRole" type="xsd:string"/>
        <xsd:element name="RelationshipToCardinality" type="xsd:string"/>
        <xsd:element name="RelationshipType" type="xsd:string"/>
        <xsd:element name="RelationshipDescription" type="xsd:string"/>
    </xsd:complexType>

</xsd:schema>
```

The XML schema definition was validated using the online schema validator available on http://www23.w3.org/cgi-bin/xmlschema-check

H.4.1 Schema validating with XSV 1.159/1.67 of 2000/09/11 12:59:09

- **Target**: `file:/usr/local/XSV/old_xsvlog/@27597.6uploaded`
  (Real name: D:\PhD\CapturedModels\EnterpriseModelsSchema.txt)
- **docElt**: `{http://www.w3.org/2001/XMLSchema}schema`
- No declaration for document root found, validation was lax
- **schemaLocs**:
- The schema(s) used for schema-validation had no errors
- No schema-validity problems were found in the target

APPENDIX I: FULL TEXT AND READABILITY STATISTICS AS REPORTED BY MS-WORD 2002.

		COUNTS				AVERAGES			READABILITY		
ModelID	DocType	Words	Chars	Para-graphs	Senten-ces	Sentence/ Paragraph	Words / Sentence	Chars / Word	Passive Sentences	Flesch Reading Ease	Flesch-Kincaid Grade Level
AIAI	Definitions	3432	20199	177	117	1.3	17.7	5.6	14%	23.8	12.0
AKMA	Definitions	3691	21161	80	211	2.9	18.1	5.2	19%	34.6	12.0
ARRI	Definitions	5073	31070	121	304	2.5	16.5	5.9	25%	17.3	12.0
BAAN	Definitions	1751	10243	169	169	1.0	13.8	5.7	0%	34.9	11.8
BelgAcc	None										
BOMA	Definitions	5226	26946	135	291	2.1	17.8	5.0	38%	42.3	11.7
BOMA	BookScan	1609	8601	22	71	4.4	22.1	5.2	35%	28.4	12.0
CYC	Definitions	50303	289670	765	2911	3.8	21.4	5.3	13%	31.8	12.0
Fowler	BookScan	2258	11556	26	135	5.8	16.6	5.0	23%	41.2	11.4
Hay	BookScan	2465	12050	53	118	3.6	19.1	4.7	27%	48.0	11.1
Inmon	None										
Japan	None										
Miller	BookScan	4908	26766	35	227	6.6	21.5	5.3	18%	23.7	12.0
NHS	Definitions	6165	33379	216	388	1.8	15.7	5.2	21%	33.5	12.0
OTTBig	Definitions	31296	166724	451	1595	3.7	19.5	5.1	24%	34.4	12.0
OTTSmall	Definitions	20823	111521	251	1034	4.3	20.0	5.2	23%	33.6	12.0
Purdue	Definitions	2180	13819	75	80	1.0	27.2	6.2	1%	1.3	12.0
Random	BookScan	3518	17199	181	489	2.7	7.1	4.5	1%	68.1	5.4
SanFran	Definitions	3142	16496	67	179	3.0	17.1	5.0	30%	37.7	12.0
SAP	BookScan	2265	13311	36	88	3.0	24.9	5.7	53%	16.9	12.0
Silverston	BookScan	2221	10823	34	109	5.1	19.7	4.7	28%	45.0	11.8
TOVE	Definitions	3225	14756	111	129	1.1	24.5	4.4	5%	53.1	11.9
USB	UserMan	6063	29395	137	368	3.4	15.9	4.6	22%	45.4	10.6
USB	ProgRef	1149	2686	24	66	3.4	17.0	4.8	43%	45.2	11.1

APPENDIX J: RELATIVE OVERLAP BETWEEN DICTIONARIES USED FOR SEMANTIC ANALYSIS

It is an interesting exercise to check the overlap between the various word lists used for much of the semantic analysis (perspicuity and genericity measures). The following two tables detail the total number of words of a dictionary that can be found in the other dictionaries.

Found in / Words from	BL	MW	OB	SB	WP	All	
BL	9863	2256	810	1371	915	9863	words
MW	2256	3700	874	1365	900	3700	words
OB	810	874	1056	764	510	1056	words
SB	1371	1365	764	1811	678	1811	words
WP	915	900	510	678	1323	1323	words
All words combined	9863	3700	1056	1811	1323	11838	words

Note that the matrix must be symmetric along its diagonal: if 810 words from OB are found in BL, then by necessity 810 words from BL must be found in OB (since they are the same words).
This can be expressed in relative fashion in the following way.

Found in / % Words from	BL	MW	OB	SB	WP	Total Size of Dictionary
BL	100%	23%	8%	14%	9%	9863 words
MW	61%	100%	24%	37%	24%	3700 words
OB	77%	83%	100%	72%	48%	1056 words
SB	76%	75%	42%	100%	37%	1811 words
WP	69%	68%	39%	51%	100%	1323 words
All words combined	83%	31%	9%	15%	11%	11838 words

E.g. 61% of the words in the Moneywords (MW) list (i.e. 2256 / 3700) can also be found in the BL list, whereas only 23% of the words in the BL list are found in the MW list (i.e. the same 2256 but expressed as a fraction of the total BL list size i.e. 2256 / 9863).
Although it is fairly straightforward to interpret the data, the different sizes of the worldlists create methodological problems. For instance, it is clear that a very large proportion (>60%) of the words in all of the dictionaries can be found in BL as indicated by the first (BL) data column. It is also clear that a roughly equally large proportion of all dictionaries' words (>68%; see second data column) are found in the MW – with the exception of BL, which is 3 times the size of MW and therefore no more than 30% of MW could ever be contained in MW. However, because of the much smaller size of MW as compared to BL, the figures in the second data column are much more significant. For instance, if one looks at the SB list of 1811, almost exactly the same number of words (1371 versus 1365), representing three-quarters of the entire SB list, can be found in both BL and MW. Does this mean that the BL and MW are equally similar to SB? Evidently not, since BL is three times as big as MW.
The problem poses itself on how to calculate the relative overlap or similarity between these wordlists. Traditional distance measures in mathematics and linguistics are based on Euclidean distances. Unfortunately, this requires that vectors of equal size be used i.e. the dictionaries would have to have the same number of words.
An interesting alternative used in linguistics to find the "shortest distance" between two text strings, which can be of unequal length, is the Hamming distance, which looks at the minimum number of individual character substitutions or changes (insertions/deletions) that are required to change one string into another. In principle, dictionaries can be seen as long strings which are concatenations of the individual words (with possibly a separator character added inbetween words). However, there are a number of problems associated with the Hamming distance.

1. It works on a character basis, whereas wordlists use words (their length is immaterial) as natural unit. It should not make a difference whether a 5 or a 10-letter word is missing, which the Hamming distance does not allow for.
2. It does not scale well when used between more than two strings of different sizes: two roughly similar strings of length 100 will have a much smaller Hamming difference than two roughly similar strings of length 1000.

3. Word meanings are not taken into account: the Hamming difference between "half" and "calf" is the same as between "organize" and "organise" and one fifth of the difference between "organize" and "organization"
4. Calculating the Hamming difference for long strings (typically more than 10 000 characters for the word lists under consideration) is computationally intensive.

Luckily, other measures have been suggested to measure the similarity between vectors of different length. Although these have been discussed in a number of publications, [DUCH00] gives a clear and mathematically precise overview of the various measures.

The tables below calculate the most popular linguistic distance measures available to measure the overlap between wordlists. Note that all display the desirable property that they are symmetrical measures i.e. the distance from BL to MW is the same as the distance from MW to BL. To reduce clutter, therefore, only one triangle of values has been given. All of them are normalized so that complete overlap results in a distance of 0, whereas no overlap implies a distance of 1.

J.1 Similarity index 1: COSINE distance

Cosine distance = 1 - sizeOverlap / Sqrt (sizeA * sizeB)

Found in Words from	BL	MW	OB	SB	WP
BL	0	0.62655	0.74901	0.67561	0.7467
MW		0	0.55784	0.47268	0.59322
OB			0	0.44754	0.56852
SB				0	0.56198
WP					0

J.2 Similarity index2: JACCARD distance

Jaccard distance = 1 - sizeOverlap / (sizeA + sizeB - sizeOverlap)

Found in Words from	BL	MW	OB	SB	WP
BL	0	0.80048	0.91987	0.86693	0.91091
MW		0	0.77486	0.67077	0.78171
OB			0	0.63671	0.72713
SB				0	0.72394
WP					0

J.3 Similarity index3: DICE distance

Dice distance = 1 - 2 * sizeOverlap / (sizeA + sizeB)

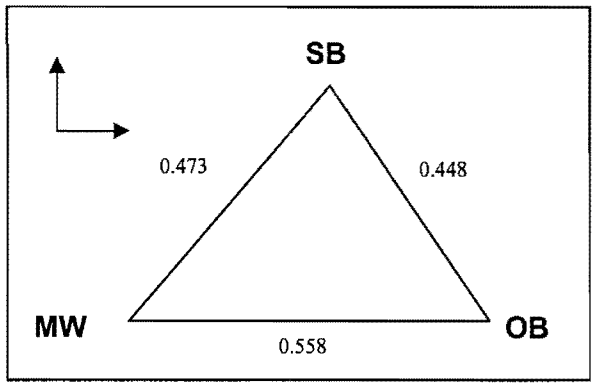
Found in Words from	BL	MW	OB	SB	WP
BL	0	0.66733	0.85163	0.76512	0.8364
MW		0	0.63246	0.50463	0.64165
OB			0	0.46704	0.57125
SB				0	0.56733
WP					0

All of the above distances are normalized so that complete overlap results in a distance of 0, whereas no overlap implies a distance of 1. If similarity indices are preferred, the distance can be subtracted from 1 (thus simplifying all of the formulas even further) and expressed as a percentage.

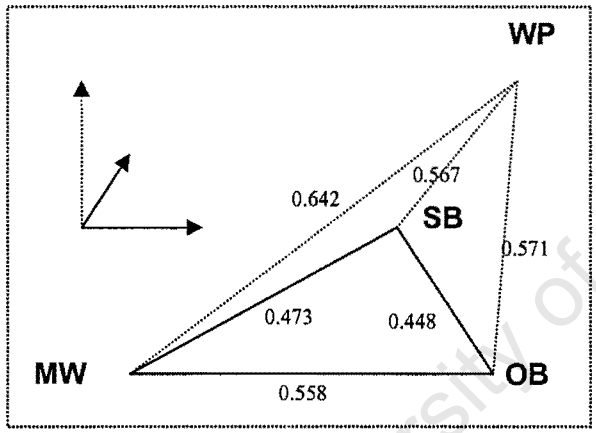
If one looks at the above tables, there appears to be no sound reason to prefer one measure over any other measure.

Whichever measure is used, it is clear that OB and SB are the most similar dictionaries, having the smallest distance between them. Almost equally similar are SB and MW. It is not surprising, therefore (though not mathematically necessary), that there is also relatively short distance between MW and OB. This triad could be visualized as the corners of a triangle on a plane with the length of each side corresponding to the respective distances between each dictionary (corner). In this case, we would have a near-equilateral triangle with the

shortest base between OB and SB as described above. The exact shape of the triangle would depend on the choice of distance measure used. The figure below illustrates the proposed visualization for the cosine distance.



It is possible to visualize the distance of this triad of wordlists to the next closest wordlist, which is WP. WP is virtually the same distance from SB as from OB but, depending on the measure used, a bit further removed from MW. This could be visualized in three-dimensional space by building a non-regular tetrahedron with the previously plotted triad forming the triangular base, where the top of the "pyramid" would be WP, which would lie approximately halfway between SB & OB, but a bit higher up than the distance between them, and with a longer vertex going to MW.



Far removed from all other four wordlists is BL. This cannot be visualized in three-dimensional space since a fourth dimension is required to plot this. However, it is clear from the tables that BL is relatively closest to MW, then to SB and furthest removed but approximately equally distance from OB and WP.
A final remark is that the choice of distance measure does affect the relative proximity between wordlists. If the cosine distance is used (chosen), then OB is closer to MW (0.5578) than it is to WP (0.5685). However, if a Dice distance is chosen, OB becomes relatively much closer to WP (0.5712) than to MW (0.6325).

APPENDIX K: TABLES FOR MODEL SIMILARITY

This appendix contains various tables with alternative overlap counts and similarity calculations. They are provided here in support of the tables provided in the main text where particular approaches or calculations were discussed.

K.1 Raw model synonym overlap counts before averaging the cell counts (i.e. average between a cell and its transposed value).

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US	List Size
AI	94	18	30	38	2	31	58	30	42	34	12	29	35	44	32	20	18	37	22	28	32	35	14	94
AK	22	83	25	39	6	33	51	24	36	50	14	21	32	37	25	21	19	38	29	34	25	25	21	83
AR	27	18	147	50	2	38	52	23	42	47	23	23	62	45	28	44	20	55	24	33	28	38	11	147
BA	54	44	67	258	10	101	139	71	106	125	31	63	93	123	91	72	46	121	74	95	91	77	61	258
BE	1	1	1	2	349	1	2	1	1	3	1	0	2	4	3	1	1	2	1	2	3	2	1	349
BO	41	33	54	84	7	174	101	49	89	91	28	38	66	86	70	56	28	95	51	73	70	55	41	174
CY	98	92	103	188	12	157	697	119	155	212	53	114	126	208	159	103	95	182	116	134	159	163	92	697
FO	33	25	33	66	4	50	67	113	60	57	11	48	41	57	43	36	29	55	40	52	43	41	27	113
HA	53	46	66	110	12	92	119	73	227	121	31	66	90	110	81	80	43	111	58	94	81	71	50	227
IN	45	59	67	135	12	100	157	65	113	411	33	56	109	186	144	89	49	118	71	111	144	72	88	411
MI	8	9	12	16	2	12	22	6	13	16	60	6	13	16	11	10	8	17	13	13	11	16	10	60
NH	37	30	34	58	2	42	68	51	63	52	19	177	42	53	39	37	31	60	36	45	39	53	26	177
NI	29	18	59	72	6	52	72	34	64	76	18	29	196	71	47	62	25	79	28	49	47	49	32	196
OB	48	39	62	118	9	93	147	56	99	174	27	48	89	451	305	70	42	104	61	95	305	69	106	451
OD	29	25	37	77	6	64	91	38	67	122	14	25	51	277	277	47	27	61	39	64	277	34	82	277
PU	17	15	42	52	4	36	50	25	49	60	14	21	62	50	33	129	10	55	23	36	33	23	25	129
RA	11	10	9	27	3	15	27	12	22	28	5	11	16	27	19	12	249	22	9	14	19	14	11	249
SA	58	51	81	127	6	100	141	70	118	124	36	68	109	116	85	84	39	262	75	99	85	76	56	262
SF	30	32	28	68	4	50	77	40	50	63	19	36	39	62	46	37	26	56	112	56	46	42	39	112
SI	37	36	40	85	5	75	88	51	87	94	22	48	60	84	66	54	33	84	55	161	66	52	46	161
SR	29	25	37	77	6	64	91	38	67	122	14	25	51	277	277	47	27	61	39	64	277	34	82	277
TO	37	19	40	66	4	49	98	36	57	61	21	46	53	67	40	31	27	59	36	38	40	333	32	333
US	17	16	18	53	1	41	59	23	42	70	11	19	29	83	73	29	18	40	33	39	73	31	129	129

K.2 Cosine similarities coefficients for models, based on synonyms.

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US
AI	100%	23%	24%	30%	1%	28%	30%	31%	33%	20%	13%	26%	24%	22%	19%	17%	9%	30%	25%	26%	19%	20%	14%
AK	23%	100%	19%	28%	2%	27%	30%	25%	30%	30%	16%	21%	20%	20%	16%	17%	10%	30%	32%	30%	16%	13%	18%
AR	24%	19%	100%	30%	1%	29%	24%	22%	30%	23%	19%	18%	36%	21%	16%	31%	8%	35%	20%	24%	16%	18%	11%
BA	30%	28%	30%	100%	2%	44%	39%	40%	45%	40%	19%	28%	37%	35%	31%	34%	14%	48%	42%	44%	31%	24%	31%
BE	1%	2%	1%	2%	100%	2%	1%	1%	2%	2%	1%	0%	2%	2%	1%	1%	1%	1%	1%	1%	1%	1%	0%
BO	28%	27%	29%	44%	2%	100%	37%	35%	46%	36%	20%	23%	32%	32%	31%	31%	10%	46%	36%	44%	31%	22%	27%
CY	30%	30%	24%	39%	1%	37%	100%	33%	34%	34%	18%	26%	27%	32%	28%	26%	15%	38%	35%	33%	28%	27%	25%
FO	31%	25%	22%	40%	1%	35%	33%	100%	42%	28%	10%	35%	25%	25%	23%	25%	12%	36%	36%	38%	23%	20%	21%
HA	33%	30%	30%	45%	2%	46%	34%	42%	100%	38%	19%	32%	37%	33%	30%	38%	14%	47%	34%	47%	30%	23%	27%
IN	20%	30%	23%	40%	2%	36%	34%	28%	38%	100%	16%	20%	33%	42%	39%	32%	12%	37%	31%	40%	39%	18%	34%
MI	13%	16%	19%	19%	1%	20%	18%	10%	19%	16%	100%	12%	14%	13%	10%	14%	5%	21%	20%	18%	10%	13%	12%
NH	26%	21%	18%	28%	0%	23%	26%	35%	32%	20%	12%	100%	19%	18%	14%	19%	10%	30%	26%	28%	14%	20%	15%
NI	24%	20%	36%	37%	2%	32%	27%	25%	37%	33%	14%	19%	100%	27%	21%	39%	9%	41%	23%	31%	21%	20%	19%
OB	22%	20%	21%	35%	2%	32%	32%	25%	33%	42%	13%	18%	27%	100%	82%	25%	10%	32%	27%	33%	82%	18%	39%
OD	19%	16%	16%	31%	1%	31%	28%	23%	30%	39%	10%	14%	21%	82%	100%	21%	9%	27%	24%	31%	100%	12%	41%
PU	17%	17%	31%	34%	1%	31%	26%	25%	38%	32%	14%	19%	39%	25%	21%	100%	6%	38%	25%	31%	21%	13%	21%
RA	9%	10%	8%	14%	1%	10%	15%	12%	14%	12%	5%	10%	9%	10%	9%	6%	100%	12%	10%	12%	9%	7%	8%
SA	30%	30%	35%	48%	1%	46%	38%	36%	47%	37%	21%	30%	41%	32%	27%	38%	12%	100%	38%	45%	27%	23%	26%
SF	25%	32%	20%	42%	1%	36%	35%	36%	34%	31%	20%	26%	23%	27%	24%	25%	10%	38%	100%	41%	24%	20%	30%
SI	26%	30%	24%	44%	1%	44%	33%	38%	47%	40%	18%	28%	31%	33%	31%	31%	12%	45%	41%	100%	31%	19%	29%
SR	19%	16%	16%	31%	1%	31%	28%	23%	30%	39%	10%	14%	21%	82%	100%	21%	9%	27%	24%	31%	100%	12%	41%
TO	20%	13%	18%	24%	1%	22%	27%	20%	23%	18%	13%	20%	20%	18%	12%	13%	7%	23%	20%	19%	12%	100%	15%
US	14%	18%	11%	31%	0%	27%	25%	21%	27%	34%	12%	15%	19%	39%	41%	21%	8%	26%	30%	29%	41%	15%	100%

K.3 Cosine similarities coefficient RANKINGS for models, based on synonyms.

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US
AI		150	138	101	260	112	91	88	70	168	212	126	143	152	181	195	234	93	129	123	181	165	209
AK	150		176	109	244	114	98	130	97	104	198	158	173	172	196	194	230	94	77	92	196	213	188
AR	138	176		95	262	106	139	153	100	145	185	191	53	162	199	82	239	58	166	142	199	192	225
BA	101	109	95		245	24	39	32	20	33	183	110	48	55	78	63	207	15	26	23	78	137	80
BE	260	244	262	245		248	253	256	243	246	258	264	249	247	251	257	261	254	255	250	251	259	263
BO	112	114	106	24	248		46	56	19	52	174	149	75	74	89	86	227	18	51	22	89	154	115
CY	91	98	139	39	253	46		66	61	60	186	125	122	76	107	128	204	44	59	67	107	119	133
FO	88	130	153	32	256	56	66		27	111	228	57	132	134	146	131	217	50	54	42	146	171	163
HA	70	97	100	20	243	19	61	27		40	184	72	49	68	102	45	210	17	64	16	102	144	121
IN	168	104	145	33	246	52	60	111	40		201	169	69	25	35	71	221	47	81	34	35	187	62
MI	212	198	185	183	258	174	186	228	184	201		220	208	215	232	211	242	157	175	190	232	214	223
NH	126	158	191	110	264	149	125	57	72	169	220		180	189	205	178	231	99	127	113	205	164	203
NI	143	173	53	48	249	75	122	132	49	69	208	180		120	159	38	235	28	151	87	159	170	179
OB	152	172	162	55	247	74	76	134	68	25	215	189	120		13	136	229	73	116	65	13	193	37
OD	181	196	199	78	251	89	107	146	102	35	232	205	159	13		155	236	117	140	84	1	218	30
PU	195	194	82	63	257	86	128	131	45	71	211	178	38	136	155		241	43	135	83	155	216	161
RA	234	230	239	207	261	227	204	217	210	221	242	231	235	229	236	241		222	226	224	236	240	238
SA	93	94	58	15	254	18	44	50	17	47	157	99	28	73	117	43	222		41	21	117	148	124
SF	129	77	166	26	255	51	59	54	64	81	175	127	151	116	140	135	226	41		29	140	167	96
SI	123	92	142	23	250	22	67	42	16	34	190	113	87	65	84	83	224	21	29		84	177	105
SR	181	196	199	78	251	89	107	146	102	35	232	205	159	13	1	155	236	117	140	84		218	30
TO	165	213	192	137	259	154	119	171	144	187	214	164	170	193	218	216	240	148	167	177	218		202
US	209	188	225	80	263	115	133	163	121	62	223	203	179	37	30	161	238	124	96	105	30	202	

K.4 Jaccard similarity between models using synonyms.

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US
AI	100%	13%	13%	15%	0%	16%	11%	18%	17%	8%	7%	14%	12%	9%	9%	9%	4%	15%	14%	15%	9%	9%	7%
AK	13%	100%	10%	14%	1%	15%	10%	14%	15%	12%	9%	11%	10%	8%	7%	9%	5%	15%	19%	17%	7%	6%	10%
AR	13%	10%	100%	17%	0%	17%	10%	12%	17%	11%	9%	10%	21%	10%	8%	18%	4%	20%	11%	13%	8%	9%	6%
BA	15%	14%	17%	100%	1%	27%	21%	23%	29%	24%	8%	16%	22%	20%	19%	19%	8%	31%	24%	27%	19%	14%	17%
BE	0%	1%	0%	1%	100%	1%	1%	1%	1%	1%	0%	0%	1%	1%	1%	1%	0%	1%	1%	1%	1%	0%	0%
BO	16%	15%	17%	27%	1%	100%	17%	21%	29%	20%	9%	13%	19%	17%	17%	18%	5%	29%	21%	28%	17%	11%	16%
CY	11%	10%	10%	21%	1%	17%	100%	13%	17%	20%	5%	12%	12%	18%	15%	10%	7%	20%	14%	15%	15%	15%	10%
FO	18%	14%	12%	23%	1%	21%	13%	100%	24%	13%	5%	21%	14%	11%	12%	14%	6%	20%	22%	23%	12%	9%	12%
HA	17%	15%	17%	29%	1%	29%	17%	24%	100%	22%	8%	19%	22%	18%	17%	22%	7%	31%	19%	30%	17%	13%	15%
IN	8%	12%	11%	24%	1%	20%	20%	13%	22%	100%	5%	10%	18%	26%	24%	16%	6%	22%	15%	22%	24%	10%	17%
MI	7%	9%	9%	8%	0%	9%	5%	5%	8%	5%	100%	6%	6%	4%	4%	7%	2%	9%	10%	9%	4%	5%	6%
NH	14%	11%	10%	16%	0%	13%	12%	21%	19%	10%	6%	100%	11%	9%	8%	10%	5%	17%	14%	16%	8%	11%	8%
NI	12%	10%	21%	22%	1%	19%	12%	14%	22%	18%	6%	11%	100%	14%	12%	24%	5%	26%	12%	18%	12%	11%	10%
OB	9%	8%	10%	20%	1%	17%	18%	11%	18%	26%	4%	9%	14%	100%	67%	12%	5%	18%	12%	17%	67%	9%	19%
OD	9%	7%	8%	19%	1%	17%	15%	12%	17%	24%	4%	8%	12%	67%	100%	11%	5%	16%	12%	17%	100%	6%	24%
PU	9%	9%	18%	19%	1%	18%	10%	14%	22%	16%	7%	10%	24%	12%	11%	100%	3%	22%	14%	18%	11%	6%	12%
RA	4%	5%	4%	8%	0%	5%	7%	6%	7%	6%	2%	5%	5%	5%	5%	3%	100%	6%	5%	6%	5%	4%	4%
SA	15%	15%	20%	31%	1%	29%	20%	20%	31%	22%	9%	17%	26%	18%	16%	22%	6%	100%	21%	28%	16%	13%	14%
SF	14%	19%	11%	24%	1%	21%	14%	22%	19%	15%	10%	14%	12%	12%	12%	14%	5%	21%	100%	26%	12%	10%	18%
SI	15%	17%	13%	27%	1%	28%	15%	23%	30%	22%	9%	16%	18%	17%	17%	18%	6%	28%	26%	100%	17%	10%	17%
SR	9%	7%	8%	19%	1%	17%	15%	12%	17%	24%	4%	8%	12%	67%	100%	11%	5%	16%	12%	17%	100%	6%	24%
TO	9%	6%	9%	14%	0%	11%	15%	9%	13%	10%	5%	11%	11%	9%	6%	6%	4%	13%	10%	10%	6%	100%	7%
US	7%	10%	6%	17%	0%	16%	10%	12%	15%	17%	6%	8%	10%	19%	24%	12%	4%	14%	18%	17%	24%	7%	100%

K.5 Jaccard similarity coefficients for models based on original entity word lists i.e. not using synonyms.

	AI	AK	AR	BA	BE	BO	CY	FO	HA	IN	MI	NH	NI	OB	OD	PU	RA	SA	SF	SI	SR	TO	US
AI	100%	5%	6%	7%	0%	8%	5%	8%	9%	4%	2%	5%	5%	5%	4%	5%	1%	6%	7%	8%	4%	5%	2%
AK	5%	100%	5%	7%	0%	8%	5%	5%	10%	7%	4%	6%	4%	3%	3%	4%	1%	7%	10%	10%	3%	3%	3%
AR	6%	5%	100%	9%	0%	8%	4%	5%	7%	6%	5%	4%	15%	4%	3%	10%	1%	11%	5%	6%	3%	5%	2%
BA	7%	7%	9%	100%	0%	16%	11%	11%	15%	15%	3%	7%	11%	11%	9%	9%	2%	19%	16%	18%	9%	8%	10%
BE	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
BO	8%	8%	8%	16%	0%	100%	8%	10%	16%	11%	4%	4%	8%	9%	8%	8%	1%	17%	13%	18%	8%	6%	7%
CY	5%	5%	4%	11%	0%	8%	100%	6%	8%	10%	2%	6%	5%	9%	6%	3%	2%	9%	7%	7%	6%	7%	5%
FO	8%	5%	5%	11%	0%	10%	6%	100%	10%	6%	0%	12%	3%	5%	6%	6%	1%	9%	14%	12%	6%	4%	6%
HA	9%	10%	7%	15%	0%	16%	8%	10%	100%	13%	2%	9%	11%	9%	7%	12%	2%	16%	9%	17%	7%	6%	7%
IN	4%	7%	6%	15%	0%	11%	10%	6%	13%	100%	2%	4%	9%	17%	16%	8%	2%	13%	8%	12%	16%	5%	10%
MI	2%	4%	5%	3%	0%	4%	2%	0%	2%	2%	100%	0%	4%	2%	2%	2%	0%	4%	3%	3%	2%	2%	2%
NH	5%	6%	4%	7%	0%	4%	6%	12%	9%	4%	0%	100%	3%	3%	2%	2%	1%	7%	6%	7%	2%	5%	3%
NI	5%	4%	15%	11%	0%	8%	5%	3%	11%	9%	4%	3%	100%	6%	5%	16%	1%	14%	4%	6%	5%	6%	4%
OB	5%	3%	4%	11%	0%	9%	9%	5%	9%	17%	2%	3%	6%	100%	61%	6%	1%	9%	7%	8%	61%	4%	13%
OD	4%	3%	3%	9%	0%	8%	6%	6%	7%	16%	2%	2%	5%	61%	100%	4%	1%	7%	7%	7%	100%	3%	16%
PU	5%	4%	10%	9%	0%	8%	3%	6%	12%	8%	2%	2%	16%	6%	4%	100%	1%	10%	5%	9%	4%	2%	5%
RA	1%	1%	1%	2%	0%	1%	2%	1%	2%	2%	0%	1%	1%	1%	1%	1%	100%	1%	0%	1%	1%	1%	1%
SA	6%	7%	11%	19%	0%	17%	9%	9%	16%	13%	4%	7%	14%	9%	7%	10%	1%	100%	11%	14%	7%	7%	8%
SF	7%	10%	5%	16%	0%	13%	7%	14%	9%	8%	3%	6%	4%	7%	7%	5%	0%	11%	100%	16%	7%	5%	10%
SI	8%	10%	6%	18%	0%	18%	7%	12%	17%	12%	3%	7%	6%	8%	7%	9%	1%	14%	16%	100%	7%	4%	8%
SR	4%	3%	3%	9%	0%	8%	6%	6%	7%	16%	2%	2%	5%	61%	100%	4%	1%	7%	7%	7%	100%	3%	16%
TO	5%	3%	5%	8%	0%	6%	7%	4%	6%	5%	2%	5%	6%	4%	3%	2%	1%	7%	5%	4%	3%	100%	3%
US	2%	3%	2%	10%	0%	7%	5%	6%	7%	10%	2%	3%	4%	13%	16%	5%	1%	8%	10%	8%	16%	3%	100%

K.6 The three most similar models when using literal entity word mapping i.e. not using synonyms

DICE similarity - closest							COSINE similarity - closest							JACCARD similarity - closest						
	Closest		2nd close		3rd close			Closest		2nd close		3rd close			Closest		2nd close		3rd close	
Model	1st	Sim	2nd	Sim	3rd	Sim	Model	1st	Sim	2nd	Sim	3rd	Sim	Model	1st	Sim	2nd	Sim	3rd	Sim
AI	HA	16%	BO	15%	FO	14%	AI	HA	18%	BO	16%	BA	15%	AI	HA	9%	BO	8%	FO	8%
AK	SI	18%	SF	17%	HA	17%	AK	HA	20%	SI	19%	SF	18%	AK	SI	10%	SF	10%	HA	10%
AR	NI	26%	SA	20%	PU	18%	AR	NI	27%	SA	21%	PU	18%	AR	NI	15%	SA	11%	PU	10%
BA	SA	32%	SI	31%	BO	28%	BA	SA	32%	SI	31%	SF	29%	BA	SA	19%	SI	18%	BO	16%
BE	OB	1%	NI	1%	BA	1%	BE	NI	1%	OB	1%	BA	1%	BE	OB	0%	NI	0%	BA	0%
BO	SI	30%	SA	28%	BA	28%	BO	SI	30%	SA	29%	BA	29%	BO	SI	18%	SA	17%	BA	16%
CY	BA	19%	IN	18%	SA	17%	CY	BA	21%	SF	20%	BO	19%	CY	BA	11%	IN	10%	SA	9%
FO	SF	25%	SI	22%	NH	21%	FO	SF	25%	SI	22%	NH	22%	FO	SF	14%	SI	12%	NH	12%
HA	SI	29%	SA	28%	BO	27%	HA	SI	29%	SA	28%	BO	27%	HA	SI	17%	SA	16%	BO	16%
IN	OB	29%	OD	27%	SR	27%	IN	OB	29%	OD	28%	SR	28%	IN	OB	17%	OD	16%	SR	16%
MI	AR	9%	NI	8%	AK	7%	MI	AR	10%	NI	9%	SA	9%	MI	AR	5%	NI	4%	AK	4%
NH	FO	21%	HA	16%	BA	14%	NH	FO	22%	HA	16%	BA	14%	NH	FO	12%	HA	9%	BA	7%
NI	PU	28%	AR	26%	SA	24%	NI	PU	28%	AR	27%	SA	24%	NI	PU	16%	AR	15%	SA	14%
OB	OD	76%	SR	76%	IN	29%	OB	OD	78%	SR	78%	IN	29%	OB	OD	61%	SR	61%	IN	17%
OD	SR	100%	OB	76%	US	28%	OD	SR	100%	OB	78%	US	30%	OD	SR	100%	OB	61%	US	16%
PU	NI	28%	HA	21%	SA	19%	PU	NI	28%	HA	22%	SA	20%	PU	NI	16%	HA	12%	SA	10%
RA	HA	4%	BA	4%	CY	3%	RA	HA	4%	BA	4%	CY	4%	RA	HA	2%	BA	2%	CY	2%
SA	BA	32%	BO	28%	HA	28%	SA	BA	32%	BO	29%	HA	28%	SA	BA	19%	BO	17%	HA	16%
SF	SI	27%	BA	27%	FO	25%	SF	BA	29%	SI	28%	FO	25%	SF	SI	16%	BA	16%	FO	14%
SI	BA	31%	BO	30%	HA	29%	SI	BA	31%	BO	30%	HA	29%	SI	BA	18%	BO	18%	HA	17%
SR	OD	100%	OB	76%	US	28%	SR	OD	100%	OB	78%	US	30%	SR	OD	100%	OB	61%	US	16%
TO	BA	15%	CY	14%	SA	12%	TO	BA	15%	CY	15%	SA	13%	TO	BA	8%	CY	7%	SA	7%
US	OD	28%	SR	28%	OB	23%	US	OD	30%	SR	30%	OB	28%	US	OD	16%	SR	16%	OB	13%

APPENDIX L: GRAPH PLOTS OF MODELS USING PAJEK.

L.1 Introduction

One of the syntactic analyses is the attempt to visualize the model networks (chapter 7). The PAJEK ("Program for Large Network Analysis) package from the University of Ljubljana (Slovenia) was found to be user-friendly yet has a very comprehensive functionality. It generates graph diagrams quickly and efficiently using a number of different layout criteria. It appears to be used quite commonly in graph research. Additional benefits are that it is available free of charge and runs on a Windows platform. It is available from http://vlado.fmf.uni-lj.si/pub/networks/pajek/.

Of the many options and network manipulations available, the following four diagrams appeared to be most productive because they produce charts that were quite distinctive and did not require the input of arbitrary parameters.

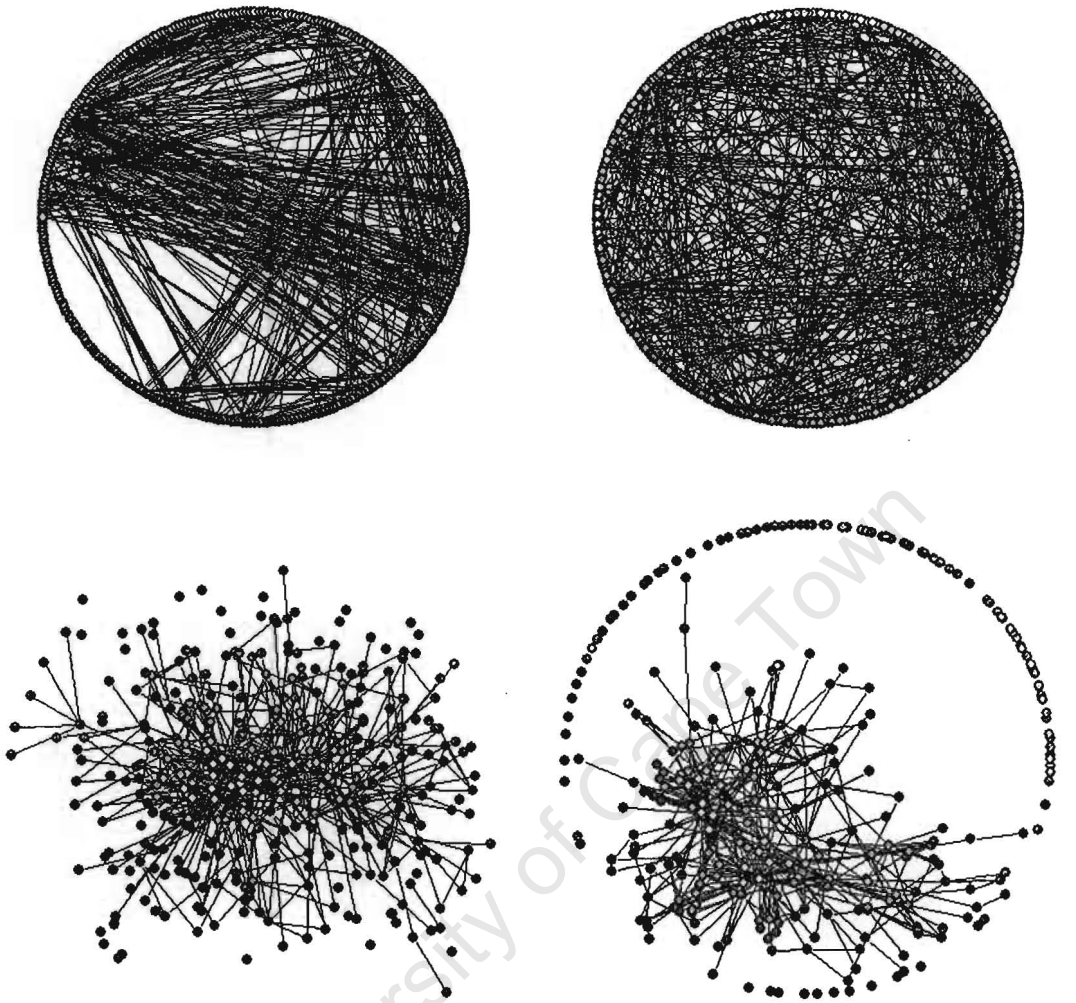
1. A plot showing entities in a circle in the sequence in which they occurred in the original source and relationships by connecting lines (the top left diagram for the figure of each model).
2. A circular plot as above, but with the entities re-arranged in random order (top right).
3. A plot with the entities positioned according to the Kamada-Kawai algorithm. This algorithm plots the enitities (little circles) on the graph so that the geometric (Euclidean) distance between them is as close as possible to the graph-theoretic (path) distance between them. It is an attempt to redraw the (n-1)-dimensional network onto two dimensions (bottom left).
4. A plots using the Fruchterman-Reingold graph layout algorithm. This is an "energy positioning" iterative procedure whereby the entities that are connected attract each other and unrelated entities repel each other (bottom right).

Pseudo-code used for the Kamada-Kawai and Fruchterman-Reingold algorithms can be found at http://repast.sourceforge.net/docs/api/uchicago/src/sim/gui/LayoutWithDisplay.html

This appendix lists examples of all four plots for those models that are similar in size: SAP, Baan, Hay, Inmon, Silverston and Random. These models were drawn using only the "domain relationship" and ignoring any structural "Is-A" relationships. Adding the latter relationships makes the diagrams even less distinct between models.

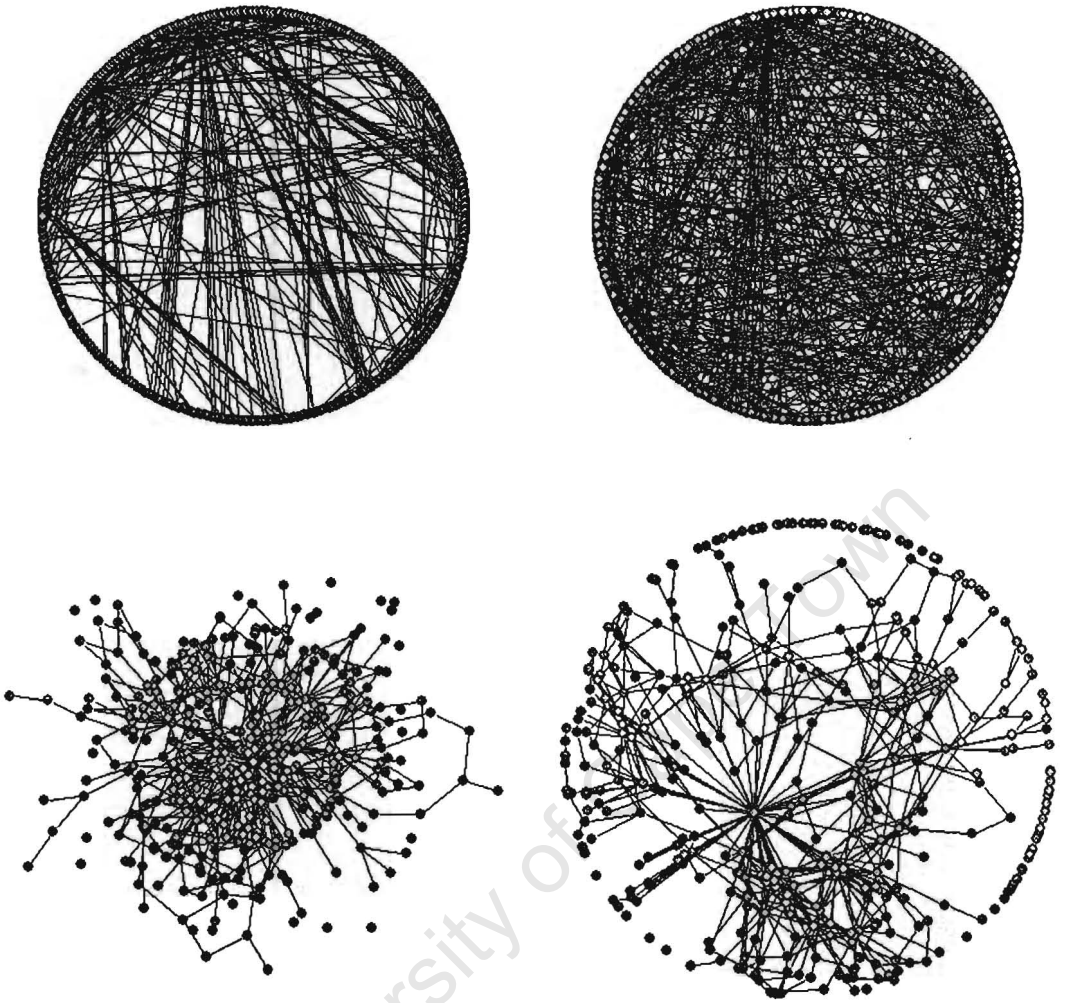
It was found that the "look and feel" of the resultant diagrams was heavily dependent on the number of nodes (entities). To illustrate the influence of the number of entities on the plot, one of the smaller models, AIAI, is also included in the appendix.

L.2 Pajek Plots of the Baan Model



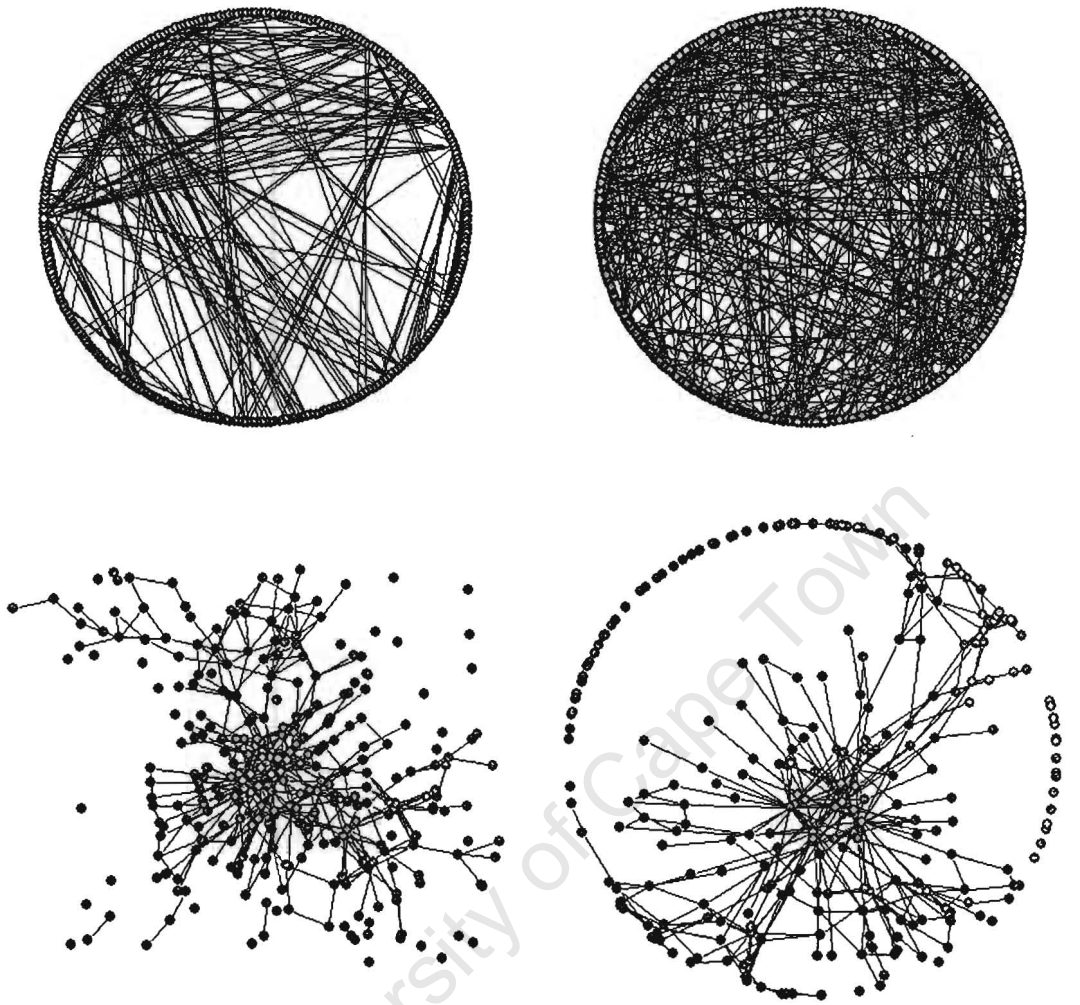
Clockwise from top left: circle plot with entities in original order; circle plot with entities in random order; plot according to the Fruchterman-Reingold graph layout algorithm; plot according to the the Kamada-Kawai algorithm.

L.3 Pajek Plots of SAP Model



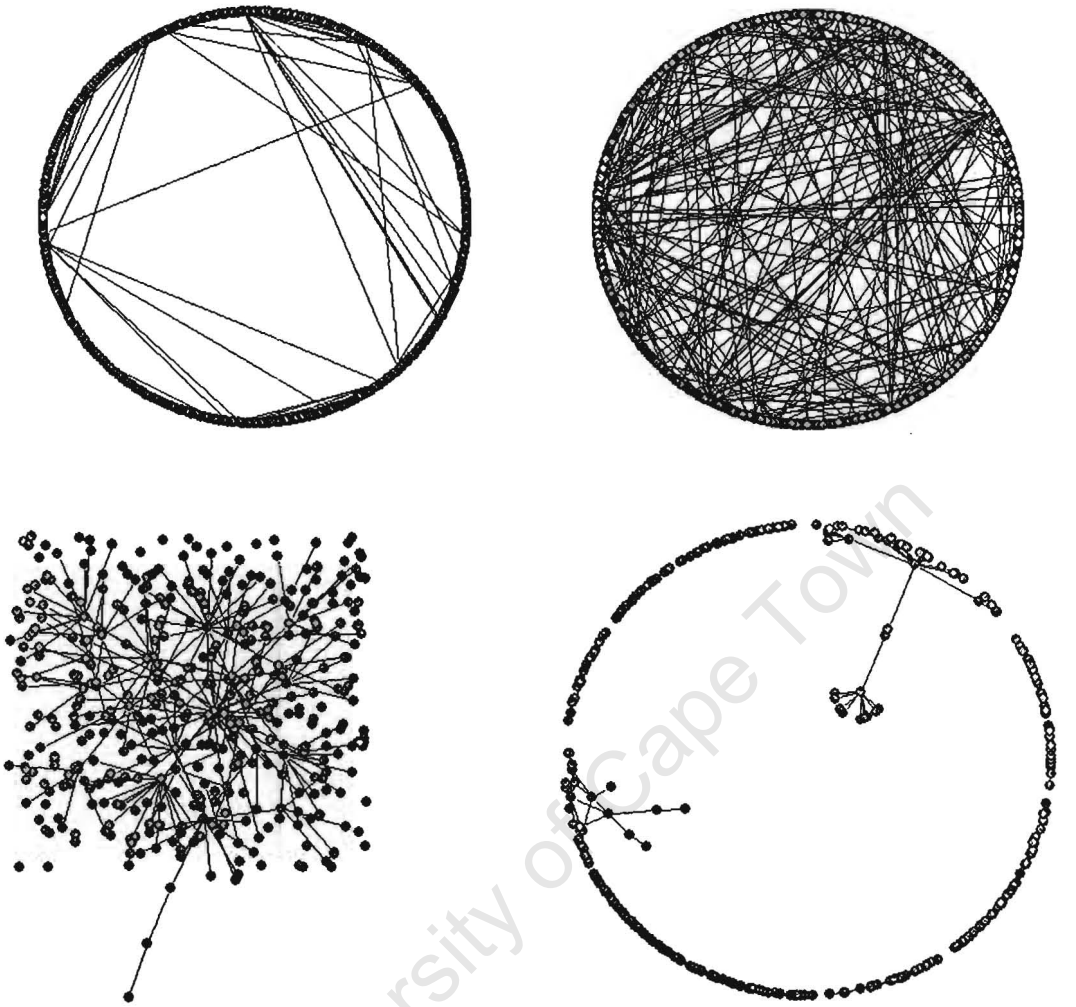
Clockwise from top left: circle plot with entities in original order; circle plot with entities in random order; plot according to the Fruchterman-Reingold graph layout algorithm; plot according to the the Kamada-Kawai algorithm.

L.4 Pajek Plots of the HAY Model



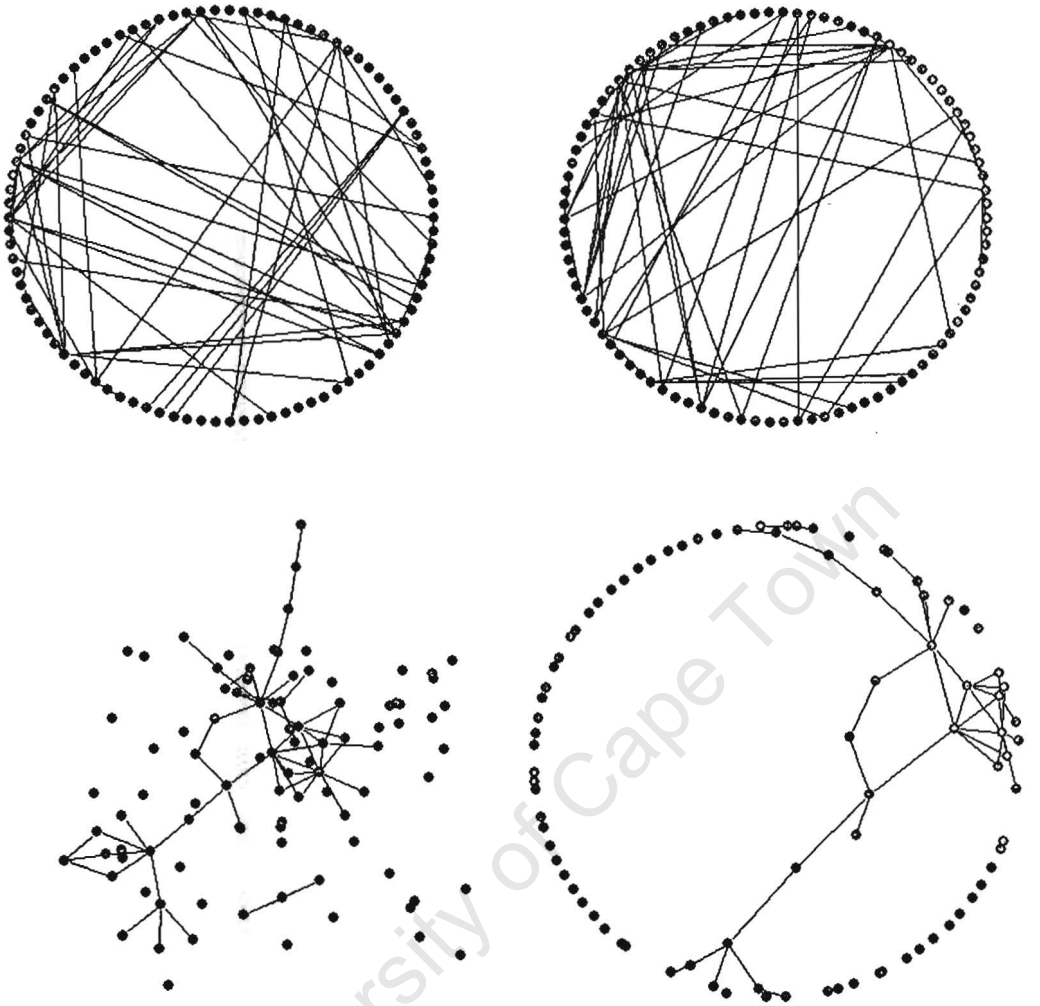
Clockwise from top left: circle plot with entities in original order; circle plot with entities in random order; plot according to the Fruchterman-Reingold graph layout algorithm; plot according to the the Kamada-Kawai algorithm.

L.5 Pajek Plots of the Inmon Model



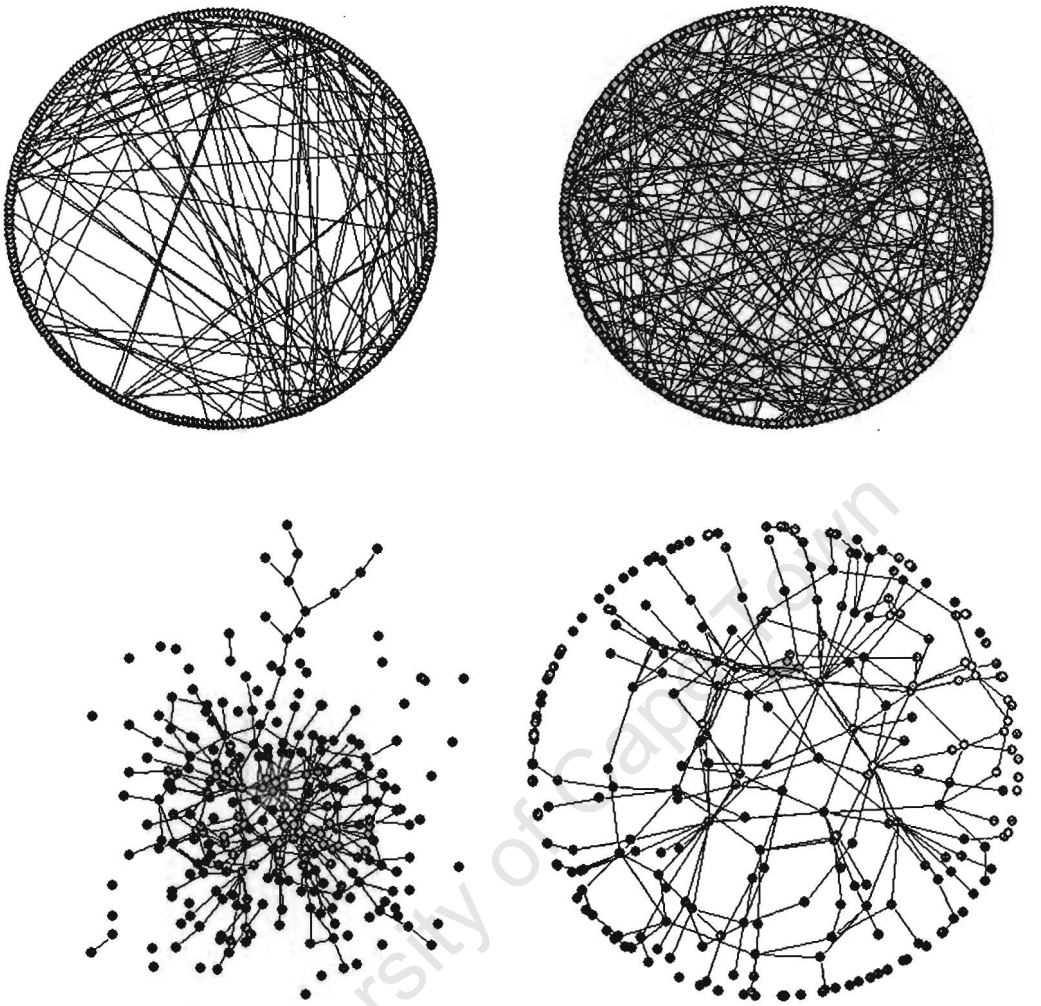
Clockwise from top left: circle plot with entities in original order; circle plot with entities in random order; plot according to the Fruchterman-Reingold graph layout algorithm; plot according to the the Kamada-Kawai algorithm.

L.6 Pajek Plots of the AIAI Model



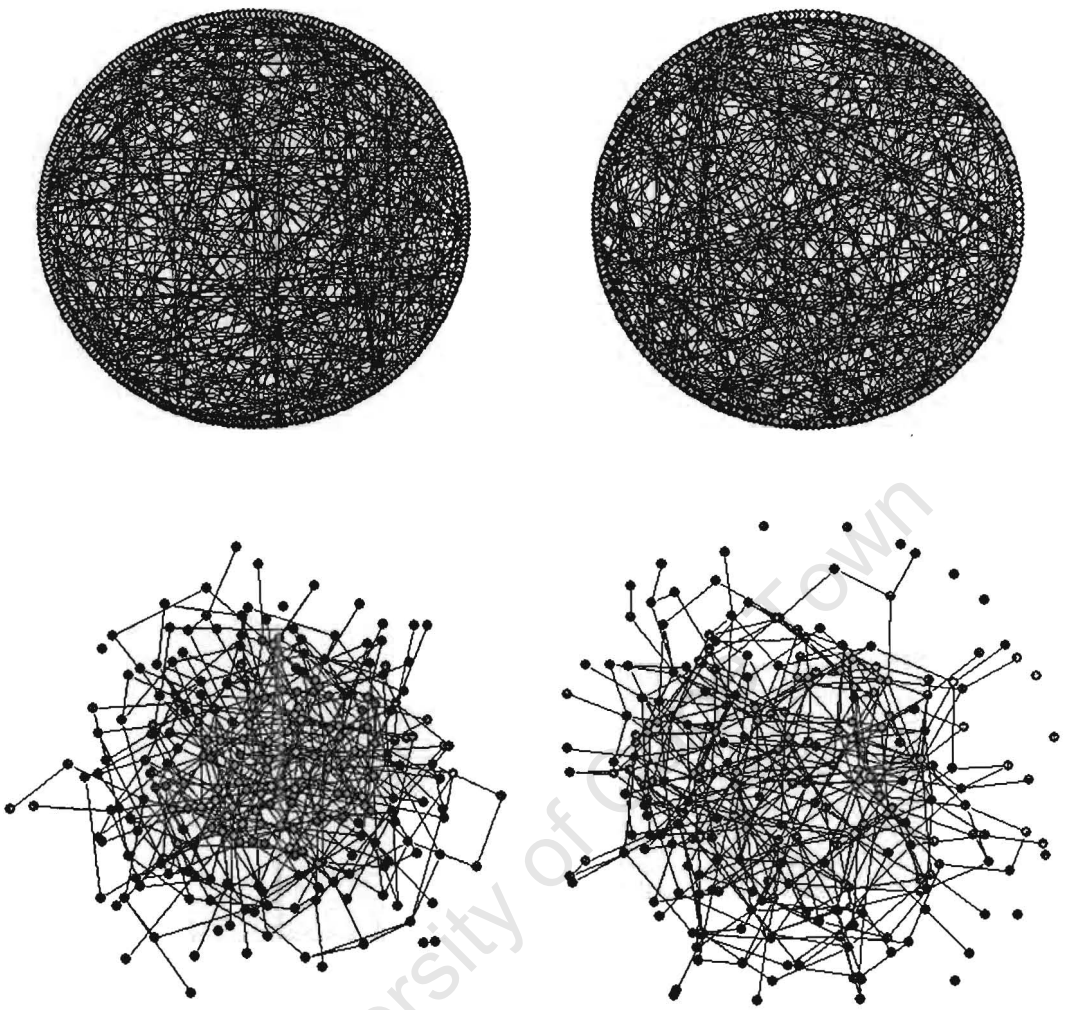
Clockwise from top left: circle plot with entities in original order; circle plot with entities in random order; plot according to the Fruchterman-Reingold graph layout algorithm; plot according to the the Kamada-Kawai algorithm.

L.7 Pajek Plots of the Silverston Model



Clockwise from top left: circle plot with entities in original order; circle plot with entities in random order; plot according to the Fruchterman-Reingold graph layout algorithm; plot according to the the Kamada-Kawai algorithm.

L.8 Pajek Plots of the Random Model



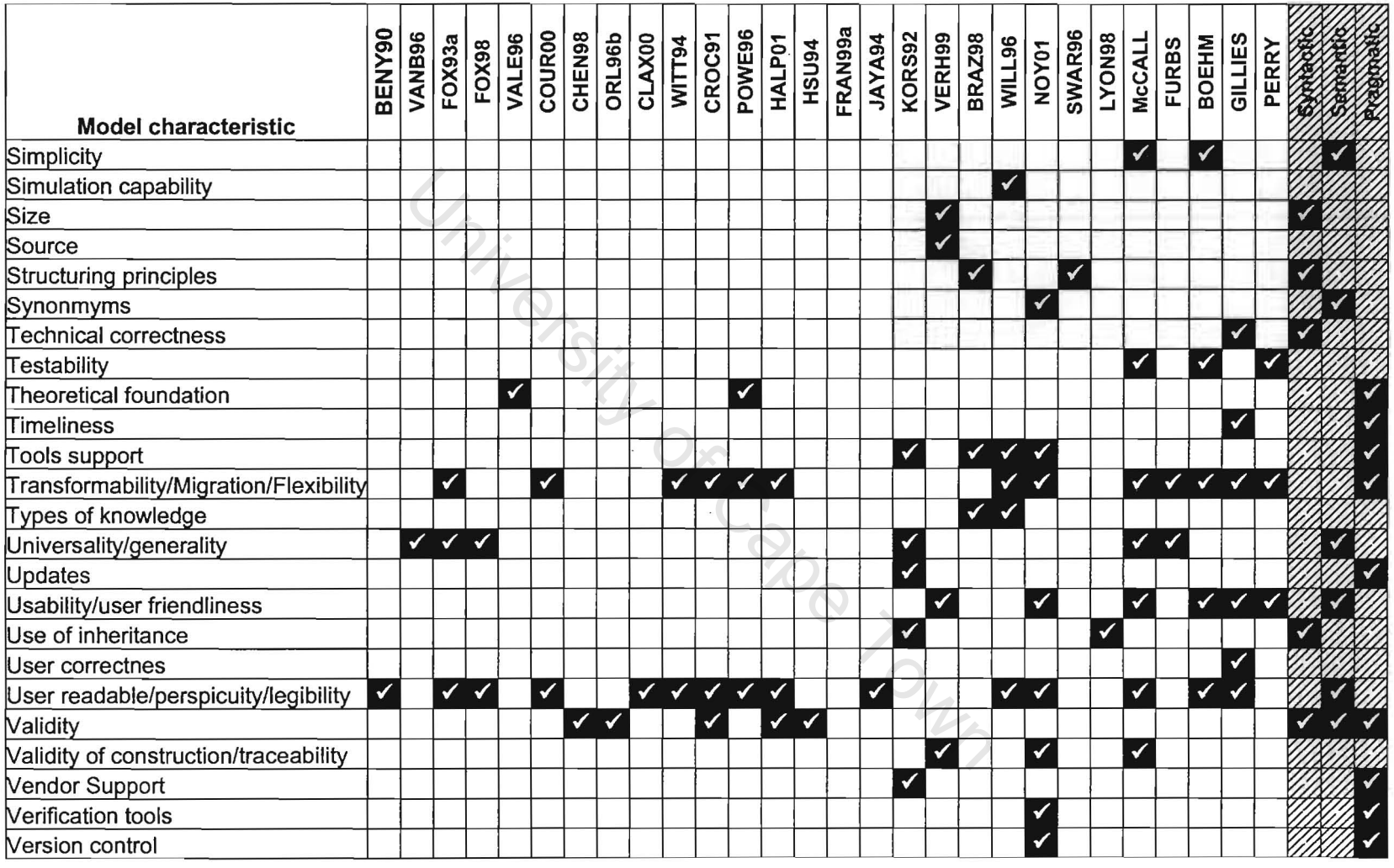
Clockwise from top left: circle plot with entities in original order; circle plot with entities in random order; plot according to the Fruchterman-Reingold graph layout algorithm; plot according to the the Kamada-Kawai algorithm.

APPENDIX M: MODEL EVALUATION CRITERIA BY SOURCE AND THEIR MAPPING TO THE FRAMEWORK.

Model characteristic	BENY90	VANB96	FOX93a	FOX98	VALE96	COUR00	CHEN98	ORL96b	CLAX00	WITT94	CROC91	POWE96	HALP01	HSU94	FRAN99a	JAYA94	KORS92	VERH99	BRAZ98	WILL96	NOY01	SWAR96	LYON98	McCALL	FURBS	BOEHM	GILLIES	PERRY	Syntactic	Semantic	Pragmatic
Abstract basic concepts					✓					✓			✓				✓		✓											✓	
Adheres to standards							✓					✓								✓	✓			✓					✓		
Aesthetics/appeal																									✓		✓		✓		
Aids clear thinking	✓																														
Alternatives/comparison							✓																						✓	✓	✓
Architecture						✓				✓																			✓		
Auditabiity																								✓		✓					
Availability						✓																									✓
Basis for communication	✓								✓		✓									✓						✓				✓	
Competent/problem oriented			✓																✓			✓								
Computer manipulation	✓		✓	✓							✓					✓														✓
Conceptual integrity										✓		✓																			
Consiseness																								✓		✓			✓		
Consistency (model)						✓	✓	✓	✓		✓			✓			✓			✓				✓	✓	✓			✓		
Consistency (representation)	✓	✓						✓			✓	✓		✓			✓							✓	✓	✓			✓		
Controlled vocabulary																					✓									✓	
Coverage/Domain (extent)						✓						✓					✓	✓	✓											✓	
Docs different levels																	✓													✓	
Docs indexed alpha/keyw/struct																	✓				✓									✓	
Docs organised & structured																	✓													✓	

Model characteristic	BENY90	VANB96	FOX93a	FOX98	VALE96	COUR00	CHEN98	ORL96b	CLAX00	WITT94	CROC91	POWE96	HALP01	HSU94	FRAN99a	JAYA94	KORS92	VERH99	BRAZ98	WILL96	NOY01	SWAR96	LYON98	McCALL	FURBS	BOEHM	GILLIES	PERRY	Syntactic	Semantic	Pragmatic
Documentation												✓					✓		✓					✓	✓					✓	
Domains & ranges																					✓										
Economics/costs																				✓							✓				✓
Effect on business/integration						✓						✓					✓					✓									✓
Efficiency (representation)			✓	✓					✓								✓			✓				✓	✓	✓	✓	✓	✓		
Efficiency/minimality (constructs)		✓		✓	✓								✓		✓					✓				✓		✓			✓		
Executability																			✓	✓											✓
Expressiveness		✓									✓		✓							✓		✓								✓	
Extensible/Customizable/Modifiable			✓			✓						✓				✓	✓			✓	✓	✓		✓	✓	✓	✓	✓			✓
Formality			✓				✓						✓		✓		✓		✓	✓											✓
Hierarchical/modular/structured	✓		✓	✓						✓		✓				✓			✓	✓				✓		✓			✓		
Human modelled explicity																				✓						✓				✓	
Implementation independence											✓									✓				✓		✓					✓
Integrity																								✓		✓	✓	✓			
Inverse relationships																					✓								✓		
Learnability/training													✓			✓	✓							✓							
Logical completeness		✓		✓	✓		✓	✓	✓						✓		✓	✓		✓				✓		✓			✓		
Loose coupling/high cohesion																	✓			✓			✓						✓		
Maintainability																								✓	✓			✓			✓

Model characteristic	BENY90	VANB96	FOX93a	FOX98	VALE96	COUR00	CHEN98	ORL96b	CLAX00	WITT94	CROC91	POWE96	HALP01	HSU94	FRAN99a	JAYA94	KORS92	VERH99	BRAZ98	WILL96	NOY01	SWAR96	LYON98	McCALL	FURBS	BOEHM	GILLIES	PERRY	Syntactic	Semantic	Pragmatic
Mappings to other vocabs, models																					✓										✓	
Maturity												✓																				✓
Methodolgy support																			✓	✓	✓											✓
Metrics provided												✓																	✓			
Model decision processes																				✓												
Multiple views																				✓	✓											
No free-standing concepts																	✓												✓			
No single childs																					✓								✓			
Outsider/user's input mechanism																		✓									✓					
Politics						✓																										
Portability																										✓	✓	✓				✓
Precision/accuracy/correctness				✓			✓	✓	✓						✓			✓						✓	✓	✓	✓	✓				
Purpose/Goal									✓							✓			✓		✓											✓
Quality						✓		✓																					✓	✓		✓
Reliability																								✓	✓	✓	✓	✓	✓			
Reusability																			✓	✓								✓				✓
Robustness (resilience to change)										✓							✓			✓		✓				✓						✓
Scalability			✓							✓																						
Scoping mechanism (zoom)	✓			✓									✓						✓													
Selfcontained/self explanatory																								✓		✓						
Semantic stability (when changed)											✓	✓	✓						✓		✓											


Model characteristic
BENY90
VANB96
FOX93a
FOX98
VALE96
COUR00
CHEN98
ORL96b
CLAX00
WITT94
CROC91
POWE96
HALP01
HSU94
FRAN99a
JAYA94
KORS92
VERH99
BRAZ98
WILL96
NOY01
SWAR96
LYON98
McCALL
FURBS
BOEHM
GILLIES
PERRY
Syntactic
Semantic
Pragmatic
Simplicity
Simulation capability
Size
Source
Structuring principles
Synonmyms
Technical correctness
Testability
Theoretical foundation
Timeliness
Tools support
Transformability/Migration/Flexibility
Types of knowledge
Universality/generality
Updates
Usability/user friendliness
Use of inheritance
User correctnes
User readable/perspicuity/legibility
Validity
Validity of construction/traceability
Vendor Support
Verification tools
Version control