DEVELOPMENT OF COMPUTER SOFTWARE SUPPORT FOR INTERPRETIVE STRUCTURAL MODELLING

BY
P.R. LAWRIE

Thesis prepared in partial fulfilment of the requirements for the degree of MSc in Operations Management

I, the undersigned, hereby declare that the work contained in this thesis, is my own original work, and has not previously in its entirety, or in part, been submitted at any university for a degree.

P.R. Lawrie ..... Date : 20/4/95

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor Tom Ryan for his advice and guidance.

INDEX

PART II - TOTAL QUALITY MANAGEMENT

PART III - THE APPLICATION OF ISM AND TQM

PART IV - REFLECTIONS

LIST OF FIGURES

LIST OF TABLES

1. SYNOPSIS

Interpretive Structural Modelling is a computer-driven method for dealing with complexity, by allowing individuals or groups to interpret the inter-relationships between different aspects or elements of a problem. In doing this interpretation, the elements are structured, and from this structuring, a graphical model is produced, allowing the modeller(s) to better understand the problem.

The main contribution of this thesis was to develop a computer package for Interpretive Structural Modelling (ISM), for use at the School of Engineering Management at the University of Cape Town. The package was developed for the IBM PC.

From the mathematical theory laid out in Warfield's book *Societical Systems - Planning Policy and Complexity*[12], a computer package was developed in Pascal. The main problem encountered in programming in Pascal was one of memory limitation due to the large matrices used by the mathematics driving the ISM process.

Once the package was complete and found to be running perfectly, a series of models were developed relating to people's perceptions of Total Quality Management (TQM) in the clothing industry. It was hoped that these models would reveal important issues on implementation of TQM in the clothing industry.

No consistent and repeatable patterns emerged from these models. Some models had large feedback sets - i.e. the aspects of TQM were seen to be very closely inter-related, while others were more linear with each aspect influencing only one or two other aspect. This led to the conclusion that there is no common thought on how best to implement a TQM initiative in the clothing industry.

An different approach to generating models was used to that conventionally used by ISM modellers. Normally models are created

in groups, with the group consensus (or vote) being used to create the model. Various researchers have written papers on the successes of group ISM work. (See refs. 17,18,19,20,21). However, it appeared that no-one had ever used ISM as a tool for comparison of thought between groups, or made models from individuals. It therefore appears that new ground was broken in that a series of models were made using the same elements set for each model, and each model was created by an individual on his own. These models were then compared one to the other.

However, in hindsight, the creation of model by individuals now appears to have certain drawbacks. Namely :

1. Different people have different definitions and perspectives on phrases.

2. The lack of debate in the creation of an individual model, which can result in :

   A. People not thinking about all relevant factors when modelling,

   B. Individuals misunderstanding the question, and as no-one is aware of their thoughts, the model is ruined by the misunderstanding.

THESIS OUTLINE

The thesis starts by looking at the subject of problems; how problems can be classified, and how ISM can help to solve problems. It goes on to give an overview of what ISM is about, and how it is useful in solving problems. A brief overview of the program from the point of view of the user is then given, followed by a brief example of what happens 'behind the scenes' during modelling.

From there, a literature review of TQM is given, looking primarily at the work of W. Edwards Deming and Philip B. Crosby.

After the literature review, the application of the ISM process in producing models on TQM is then discussed, and conclusions are drawn. A chapter of reflections on the various aspect of the thesis ends the main body of the thesis.

The details of the mathematics of ISM is given in appendix A. Appendix B explains more about the layout of the program, and how the various problems encountered in programming were overcome, as well as the bench-marking of the package to check that it was functioning correctly. Finally, appendix C contains a complete listing of the Pascal source code.

2. INTRODUCTION

The problems facing South Africa and business can no longer be dealt with by traditional problem solving approaches - the issues are too large and too complex (See Warfield[28] and Checkland[32]). A new approaches are needed. This section argues that Interpretive Structural Modelling (ISM) is such an approach.

By way of introduction to complex problems, the introduction reviews some of the economic challenges facing South African industries. Following this, there is a discussion on complexity, and how to classify problems, before going on to discuss how one might set about solving complex problems.

2.1. BACKGROUND

Companies today face many challenging problems. There are still unanswered questions on how to optimise a production line, how to keep management accounting records, how to ensure deadlines are met on budget, and how to motivate personnel to work more productively. There are many theories on these subjects, but often their application does not give the desired results.

Problems are becoming larger and more complex, especially for those companies that seek to be 'global players'. As distance and national boundaries become less of a factor in business competitiveness (due to faster transport and the trade agreements such as GATT), companies are facing an increasing amount of competition. (This is well illustrated by the initial struggles of the American motor industry to compete against imported Japanese vehicles.)

South Africa has been welcomed back into the international community following the recent first democratic elections. With this she is faced with increased competition on the one hand, and increased marketing opportunities on the other. *The World Competitiveness Report*[31] ranks South Africa as 40th in

protectionism, and 27th in overall productivity (out of 41 ranked countries). This ranking shows that South African industry has enjoyed highly protected markets, while suffering from low levels of productivity. With the signing of the GATT agreement, South African industry has a serious disadvantage in global trade, exacerbated by a ranking of 41st in skilled labour. South African industry is therefore facing both a major challenge from foreign imports, as well as a great opportunity now that the trade embargoes have been lifted.

The questions of how to develop local competitiveness and how to penetrate new markets is not one that is easily answered. There are many inter-related factors (such as culture, education and consumerism) that need to be considered before any plan of action can be drawn up.

Besides industrial problems and opportunities, South Africa is suffering from many after-effects of apartheid. The aim of the Reconstruction and Development Program (RDP) is to address these issues.

The RDP has six underlying principles (ref. 34 pp 4-12) :

1. To be an integrated and sustainable program.
2. To be a people-driven process.
3. To give peace and security to all.
4. To build a nation.
5. To link reconstruction and development.
6. To democratise South Africa.

The RDP consists of five key programmes :

1. meeting basic needs.
2. Developing our human resources.
3. Building the economy.
4. democratising the state and society.
5. implementing the RDP.

It can be argued that the RDP faces both enormous opportunities as well as enormous problems. The RDP faces the enormous problems of improving housing, health care, education, and reducing unemployment, to mention but a few. The opportunities for the RDP are to help build South Africa into the major economic power of Africa, and into a first-world country.

The *World Competitiveness Report*[31] ranks South Africa last in 'educational system', and 37th in 'manufacturing'. To achieve social development one needs economic development to create wealth to spend. However, without social development and education, economic development is hindered. The two therefore work hand in hand, and both are currently poor by world standards.

Many inter-related factors must be considered before determining how best to spend money allocated to the RDP. The problem is how to consider all the factors.

2.2. A PERSPECTIVE ON PROBLEMS

In order to solve problems, it is often useful to classify them. Mathematics teaches one to first see whether a problem is one of geometry, vectors, algebra, calculus etc. This classification helps one to reach a solution faster. The same may be said of problems, but one must first have some method of classification.

In the *Open University Handbook* for *Managing in Organisations*[16] problems are divided into two categories - difficulties, and messy problems (messes). A summary of the main differences between messes and difficulties is shown in table 2.1. A full description of the two follows.

DIFFICULTIES	MESSES
Bounded	Unbounded
Known or knowable solution	Unknown or unknowable solution
Limited timescale	Longer, unknown timescale
Priorities clear	Priorities called into question
Limited number of people involved	Large number of people involved
Know what needs to be known about the issue	Do not know what needs to be known about the issue
Can be separated out and treated as a separate matter	Cannot be disentangled from its context.
Clear definition of the problem	Difficult to make a clear definition of the problem

Table 2.1. Major differences between difficulties and messes (adapted from ref. 16.)

2.2.1. DIFFICULTIES

Difficulties are problems that are irritations. Solutions are needed, but they are usually fairly easy to find, as the problems are reasonably well defined, involves a relatively small number of people, have a limited number of solutions, and are not that costly to correct.

Difficulties encompass such things as the correct parts not getting from the store to the production line on time, a tenant not paying his rent on time, a machine that keeps breaking down, or a delay in the arrival of spare parts for a machine.

If we consider a part that was late in getting to the production line when it was available in the store all the time, we can define it as a difficulty, and not as a mess for the reasons shown in table 2.2.

DIFFICULTIES	
Bounded	Problem lies within the company
Known or knowable solution	Solution is to prevent re-occurrence.
Limited timescale	It will require a few hours to find the reasons for the problem, and a few days to implement the changes
Priorities clear	The material must arrive on time
Limited number of people involved	Only the planner, supervisor and store-keeper involved
Clear definition of the problem	The part was available, yet did not arrive at the production line in time.

Table 2.2. An example of a difficulty.

2.2.2. MESSY PROBLEMS OR MESSES

Messes are more challenging problems for several reasons : The are 'unbounded' and 'fuzzy' and are not easily defined, so a solution is not easy to come to. More people tend to be involved, and often aspects of the mess interlock. It takes a long time to fully understand the problem fully and implement some kind of 'solution'. A 'solution' to a mess often depends on the assumptions made in arriving at the 'solution', as well as the priorities one has in arriving at that 'solution'. Often people can only see their perspective of what the problem is, which make discussing the problem much more difficult.

MESSES	
Unbounded	The whole country is involved, and events both internal and external to the country will impact on the RDP
Unknown or unknowable solution	No-one can give a 'solution', but many different options will give improvements
Longer, unknown timescale	It will take years to undo the effects of apartheid, and no-one can accurately estimate the time period
Priorities called into question	Questions about the relative importance of health-care, housing, jobs and education must be answered.
Large number of people involved	The co-operation of nearly 40 million people is required.
Difficult to make a clear definition of the problem	To produce a definition of all the RDP entails results in a substantial document.

Table 2.3. An example of a messy problem.

South Africa's RDP is a very good example of a messy problem, as can be seen from the summary in table 2.3. Is education, health, housing or employment the most pressing issue? Does it vary in different locations? Should the regional governments decide on how best to spend RDP funds in their areas, or should it be done nationally? Should each region be given the same amount of money, or do some regions have better infrastructure than others?

Other examples of a mess include such situations as a company losing market share to the opposition, a company trying to improve the quality of manufacture, or someone making a decision on which new network to install in a business.

2.3. DEALING WITH PROBLEMS

There are different approaches offered on the subject of dealing with complexity. Peter Checkland puts forward an approach in his book *Systems thinking, systems practice*[32]. Likewise Flood and Carson in *Dealing With Complexity*[33]. However, here the approach of Warfield[12,28] will be examined. It has been recognised as an area needing research at the School of Engineering Management at the University of Cape Town.

In his book *Societical systems*[12], Warfield recognises that societical problems are complex, and 'are highly inter-disciplinary or transdisciplinary' are 'unbounded', and that 'important interactions among problem elements need to be considered when attempting to solve problems' (page 5).

He also states 'For example, complexity implies that one individual can, at best, do no more than contribute toward a solution. Second, complexity implies that the number of elements involved in a problem is large, and that there are many linkages among the elements stemming from a multiplicity of important contextual relations. Third complexity feeds on itself. As the complexity of a problem induces many minds to be applied to it a new complexity arises - sharing among these many minds the products of each other's thoughts and actions.' (ref. 12 page 194)

It can be seen that Warfield recognises many of the elements of messy problems as classified previously in this section. The lack of boundaries, the lack of easy 'solutions', then number of people involved, and the fact that they cannot be removed from their context. Warfield goes on from here to offer how Interpretive Structural Modelling (ISM) can be used to help deal with messy problems.

He states 'one purpose of this book is to demonstrate that it is possible to enlarge the class of activities that can be dealt

with formally, thereby diminishing the class of activities that have to be dealt with intuitively because of the absence of an alternative.' (ref. 12 page 196). Warfield notes that part of complexity is due to the quantity of information that has to be considered. He argues that the use of computers will deal with the aspect of the quantity, while the people do the creative thinking needed to solve the problem. (ref. 12, pages 196-197.)

Interpretive Structural Modelling does exactly that. It allows an individual or group to interpret a situation from their perspective(s), while a computer deals with the magnitude of the number of relationships. This interpretation is done by structuring the different elements that make up the various aspects of the problem, and the end result is a computer generated graphical model of the relationships between the different elements. The models are produced using specific rules and techniques grounded in mathematical theory. It is a method of dealing with complexity, and is not limited by the nature of the problem.

The basic method of ISM is to identify the problem that needs to be solved, and what kind of relationship is being sought between the different aspects of the problem. (e.g. The problem may be limited finance, and the relationship is one of priority - which project is most important?) A group of people who are stake-holders in the problem, or are knowledgable in the particular area being addressed are gathered together. In a brainstorming session they write down all the different aspects that pertain to the problem, each aspect being called an element. These are then worded and discussed so that everyone has the same understanding of what is meant by each element. The modelling then proceed by looking for the existence of the relationship between every possible combination of the elements. (Either the relationship exists or it does not.) Using the prescribed rules, answers are implied, thus reducing the number of questions that require answers. Once all the questions have been answered, the information is processed to produce a digraph (directed graph)

that is a model of the problem.

The only drawback is that the mathematics is of such a magnitude that a computer is needed together with the associated software in order to use the method.

PART I

AN OVERVIEW OF THE ISM PROCESS

3. OVERVIEW OF ISM

3.1. DESCRIPTION OF ISM

ISM is a methodology that helps people to deal with complexity - namely dealing with messy problems. It is a modelling method, where relationships between different elements are found. The final result is a graphical display of the different elements and their inter-relationships.

Once the subject of the model has been decided, a brainstorming session is held to determine a list of the different factors that are involved in the issue. From this list, those factors that are most relevant to the issue are selected, and placed into an element set. Also involved in the process is a relationship (chosen by the modellers) between the two elements. The process is computer driven, and consists of asking questions as to whether the different elements have the chosen relationship between them. Once the modelling is complete, the model is displayed graphically.

Moore (ref. 13, page 78) says that 'Interpretive Structural Modelling permits a group to interpret, structure, and model ideas. The method is *interpretive* in that the group's judgement decides whether and how items are related. It is *structural* in that, on the basis of the relationships, an overall structure is extracted from the complex set of items. And it is *modelling* in that the specific relationships and overall structure are portrayed in graphic form.' This interpretive nature of ISM means that even given the same elements and the same relationships, different groups can have different models. Given only the relationship and the subject matter, different groups can make completely different models consisting of very different elements and relationships.

ISM was developed by J.N. Warfield in 1973 as a methodology for dealing with complex issues. It uses discrete (or finite) mathematics to speed the modelling process. Because of the size of the mathematical manipulations, ISM sessions are done with the aid of a computer. The mathematics is designed to imply answers and thus reduce the number of questions asked. The computer is necessary because the mathematical manipulations are very large, and would be cumbersome and time-consuming to do manually.

The ISM process uses an element set, and a relationship (or contextual relation) between the elements. The relationship is typically of the form 'will lead to', 'precedes', or 'is less important than'. The elements consist of factors, or objectives relevant to the situation being modelled. So, if one element is 'laying the foundation,' another element is 'roofing the building', and the contextual relation is 'precedes,' one will be asked 'Does laying the foundation precede roofing the building?' All questions are asked in the same format of 'does element A have the chosen relationship with element B?'

All questions are answered either 'yes' or 'no'. As ISM is generally a group process, each question is first discussed before it is answered. In the event of the group not being unanimous on its choice of answer, a vote is taken, and the result of the vote is entered as the group's answer. The answer is stored as a '1' for a 'yes', and a '0' for a 'no' in a matrix, indexed by the elements. (See figure 3.1).

In addition, the modeller may enter a context in which the model is to be considered. For instance, one may consider modelling marketing strategy. However, one may wish to consider marketing strategy in the context of a booming economy, for the motor vehicle industry, or for luxury motor vehicles in a booming economy.

A-E = Element Set
1 = relationship
0 = no relationship

	A	B	C	D	E
A	1	0	0	0	0
B	1	1	0	0	0
C	1	1	1	0	0
D	1	0	1	1	0
E	1	1	0	0	1

Figure 3.1 - An example of an ISM matrix

Using Boolean algebra, ISM uses each answer to imply other answers. For example, if A enhances B and B enhances C, by implication A enhances C, and thus this question is never posed to the modellers, but is filled in automatically by the mathematics driving the process. This implication of answers results in the modellers having to answer approximately 30-40% of the total possible questions (Janes[20]). This constitutes a considerable time saving when one considers that a thirty element ISM has 900 possible questions ($30^2 = 900$) and only approximately 300 will have to be answered.

Once all the possible questions have been answered, the model may be viewed. This is done by means of directed graphs or digraphs. (See figure 3.2.) This is a very powerful method of communication that allows one to see both the element description, as well as the relationships between the elements. The arrows between elements show the direction of the relationships, and are known as paths. Thus figure 3.2 shows that : A enhances B which enhances C, which in turn enhances G. Also, D enhances E, and E enhances D. This is known as a *cycle set* or *feedback set*. The feedback set enhances F, which enhances G.

The elements in a digraph are said to be at different *levels*. The elements on the higher levels influence the elements on the lower levels (though obviously only if the two are connected). These levels are shown in figure 3.2. This displaying of levels makes it easy to see the root causes, or the points of maximum leverage where effort should be placed in solving a problem.

3.2. DIFFERENT TYPES OF MODELS

ISM has a variety of structures for aiding to understanding of complex situations. As was mentioned in section 3.1, ISM consists of elements linked together by a contextual relation. The nature or form of this contextual relation changes the structure of the ISM. This changing of structure does not refer to the layout of the digraph, (which is determined by the input of the modellers) but refers to the type of information displayed in the digraph. The three most commonly used structures follow. From these it will become apparent how broad the applications of ISM are.

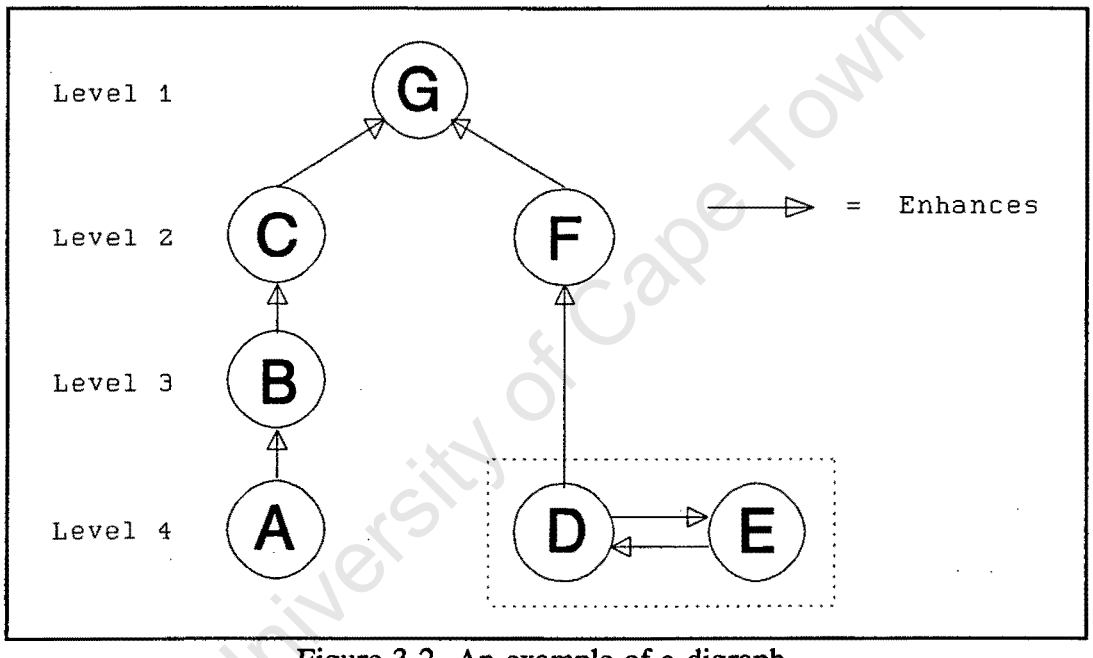


Figure 3.2. An example of a digraph.

3.2.1. *Priority Structures*

These structure are used to rank elements in **ORDER OF IMPORTANCE OR PREFERENCE.** They can be used to rank a set of projects, where budgetary constraints will only permit some to be done, or to prioritise a set of objectives for a company. The contextual relation may take the form of 'is less important than', or 'is of greater urgency than'. The end result is a digraph showing the elements in order of importance, and is useful for budget cutting, and for project management, where the order in which work can be done is of utmost importance to prevent chaos on a building site.

Warfield[12] gives an account of how a town council in Iowa used ISM to build a priority structure for their capital improvement projects scheduled for the next five years. This was viewed as successful by those involved. Saunders[21] shows how ISM was used in the development of a computer package, by using the relationship 'This aspect.... is more critical than that aspect....' From this, the group was able to decide how to write the package to best meet the user's needs.

3.2.2. *Intent Structures*

Intent structures are used to show **RELATIONSHIPS BETWEEN OBJECTIVES.** Typically the contextual relation would be of the form 'would help to achieve' or 'supports'. These structure help to show a **MEANS TOWARDS AN END.** It is therefore a useful structure for goal setting and long-term planning, where it shows what objectives must be reached first in order to reach the later objectives. This helps to focus a company's efforts in achieving those goals that will best help to achieve its longer-term policies.

Jeffrey et al[17] did both a priority structure, as well as an intent structure for a speech therapy centre. From these structures decisions were made resulting in benefits including

greater job satisfaction, increased funding, and reduced workload.

3.2.3. *Attribute Enhancement Structures*

These structures show the relationships between a set of factors, opportunities or problems. One might see 'contributes to' or 'enhances' as the contextual relation. The structure is **CAUSAL**, and the output is **DIAGNOSTIC AND AIDS UNDERSTANDING OF A SITUATION.** It is a very useful structure for problem solving, as it identifies the underlying relationships between different factors. The digraph helps to identify root causes, so time is not wasted dealing with the effects of the problem.

The structure may well be used in analysing reasons for the lead time of manufacturing being so high. It may also be used in analysing factors that are preventing a company increase its market share.

Saunders[21] gives an example of how an attribute enhancement structure was used for a redesign of an organisation, using the contextual relation 'aggravates'. From this, the group was able to see the real cause of their problems, (a lack of leadership), and address this issue head-on.

As can be seen, ISM is therefore useful for decision making in that it can be used to prioritise a list of objectives, show inter-relationships between objectives, and produce cause-and-effect models of situations.

3.3. THE PROCESS OF MODELLING.

Several steps are involved in successfully building an ISM. The steps may vary slightly depending on the situation encountered, but typically encompass the following :

3.3.1. IDENTIFICATION OF THE ISSUE.

Before a model can be made, it is necessary to identify fairly clearly what issue is to be modelled. An organisation may find that its market share is decreasing, and want to structure the factors that influence their market share in order to better understand where to begin in order to reverse the trend.

3.3.2. DECIDING ON THE TYPE OF ISM.

It is necessary to decide on the type of ISM to be constructed before the element set can be generated. This is because the type of ISM chosen helps to determine the form of the elements, as well as the wording of the relationship between the elements. Often the identification of the issue will reveal the type of ISM that should be chosen.

So, while a priority structure may help budget cutting, it will not really be of use in trying to improve market share. A priority structure may be a useful in helping to identify which market is best suited to a product. This may be done by prioritising what is important to different market sectors, and then comparing the products features, with the various market sectors to identify a good target market.

Should a company be losing its market share, and desire to do and ISM on this issue, the following question must first be answered : Is the ISM to be made to find the reason why the market share is dropping, or to help regain what has been lost? To model the cause for a decreased market share, a attribute enhancement structure should be done, while to model

how best to regain the market share is done with an intent structure.

3.3.3 SELECTING THE FACILITATOR AND GROUP

The facilitator is best described as the person who 'chairs' the ISM session. Ideally (s)he should be someone who is familiar with the ISM process and the computer software being used. (S)he needs to have experience in managing groups and group dynamics.

Her/his role includes briefly describing ISM to those members of the group not familiar with or new to the process, and guiding the process of ISM as outlined in the steps proceeding this one.

(S)he should not be part of the modelling group as this will split her/his concentration, and may result in her/him biasing the discussion toward his/her viewpoint by not allowing group members with viewpoints different to her/his own to speak.

The group itself should be selected from those people who will be involved in carrying out any decisions reached from the model. Thus it should consist of the relevant decision makers and stakeholders (i.e. Those people most effected by the decisions taken). They should obviously also have knowledge relevant to the issue.

Groups should also not be too large as the time spent on discussion grows rapidly the larger a group gets. Janes[2] recommends no more than eight, and points out that the number of possible communications between individuals in a group of n people is n(n-1). Thus increasing a group from five to six results in a 50% increase in the number of possible communications.

3.3.4. GENERATING THE ELEMENT SET

In some cases the element set may be defined, and this step merely involves phrasing them well. This is true when ranking a group of proposed projects, when only a limited number of projects can be done due to financial constraints. However, in most circumstances it will be necessary for the group to first generate the element set.

In order to generate the elements, it is advantageous to use structured methods of idea-generation. Both *Nominal Group Technique* (*NGT*) and *Ideawriting* have been found to work well within the ISM sessions. A brief summary of these two techniques is given below :

3.3.4.1. Nominal Group Technique

NGT is the most commonly mentioned process for generating the element set within the ISM literature. Warfield has described NGT as 'an efficient method of generating ideas in groups, for clarifying the generated ideas, and for developing a preliminary ranking of the set of steps.'[28] NGT may be summarised as consisting of five basic steps :

A. Formulate and test the trigger question designed to elicit the elements.
B. Silent, independent generation of ideas in writing in response to the trigger question.
C. Round-robin recording of the ideas generated on a flip-chart.
D. Serial discussion of the ideas to enhance clarity and edit the ideas.
E. The ideas are ranked in order of importance by means of voting.

3.3.4.2. Ideawriting

This technique is particularly useful both for idea generation, as well as making general ideas more specific. The ideawriting process may be summarised in six steps :

A. Formulate and test the trigger question designed to elicit the elements.
B. The large group is divided into working groups of three to six members.
C. Each participant writes down their initial response to the trigger question.
D. Each participant in the group writes down a response to what is written by the other members of the group.
E. Each participant reads the reactions, and as a group discusses the principal idea that emerges from what was written and writes it down.
F. The ideas are ranked in order of importance by means of voting.

Once the generation of potential elements is complete, a substantial list of elements may be available for modelling. Due to time limitations, it may be necessary to use only the most important twenty or thirty ideas for modelling.

It can be seen that the generation of the element set is not a quick process, and may in fact take several hours to do.

3.3.5. COMPLETE THE MODELLING

This is where the ISM computer software is used to pair the elements and pose the questions to the group. The process is continued until all questions have been answered, and the matrix is complete.

Modelling is not a quick process. The author's experience is that a group of four can take one-and-a-half hours just to answer all of the questions in a fifteen element ISM. Moore (ref. 13, page 100) states only that 'it takes a substantial amount of time for a group to consider a large number of elements during an ISM session.'

Warfield (ref. 28, page 228) give the most insight into the time taken for modelling. His data shows that for an average of 22 elements, an average time of 3.1 hours was taken. The sample size used to obtain this is not stated, and nor is it stated wether relates to the modelling process from start to finish, or just to the answering of the questions. None-the-less, it gives an indication of the time required for completing an ISM.

3.3.6. DISPLAYING THE ISM

Once the modelling is complete, the multi-level digraph can be displayed as was shown in figure 3.2. Ideally, the digraph should not merely consist of the element numbers and the links between the elements, but the full description of the element should be displayed. This ensures a more powerful and understandable display, where no reference to a separate list of elements is necessary.

3.3.7. DISCUSSING AND AMENDING THE MODEL

Once the model is displayed, the facilitator should take the group through the model to ensure that it is clearly understood. The group should then be allowed to express their views on the model. The participants may propose changes to the model. (These are generally of a minor nature.) Any changes should be carefully discussed before they are made, and it may be useful to go back to the record of 'yes' and 'no' votes made by the group. Only if there is a strong desire

to change the model should it be amended, as the model is the result of prior group consensus.

It can be see that ISM is not a quick process. It takes a group of people who are dedicated and committed, and prepared to set aside a substantial amount of time. It is therefore worthwhile taking time to consider what is the issue to be modelled, and what is the best type of ISM structure to use for modelling. More time spent considering these two factors in the beginning can greatly enhance the usefulness of the end product.

The model itself is very useful, in that it identifies the root problems, or highest priority items of the subject modelled. Because models aid understanding, decisions can be made from the structure as to how to start solving problems. It helps managers explain decisions to their subordinates when implementing decisions. It is also a very useful reference for future reviews of how the situation has changed, and whether the correct steps were taken.

3.4. THE STRENGTHS OF ISM

As has been mentioned, people face messy problems as part of everyday life. ISM has is a powerful tool, useful for dealing with these messy problems for the following reasons :

3.4.1. MENTAL LIMITATIONS

Warfield (ref. 12, page 15) notes that in dealing with complexity, one encounters mental limitations. Simon (ref. 29) believes that the short-term memory can only recall approximately 5 to 9 pieces of information (or chunks). Miller (ref. 30) concluded that the limit was 7±2 chunks. However, when dealing with three different factors, and the relationships between them, one is encountering nine 'pieces' of information. By the time one is dealing with ten factors,

and the relationships between them, one is encountering 100 pieces of information.

ISM is therefore a useful methodology in that it is able to break down complexity to manageable pieces or "chunks". This is done by focusing on only two elements at any point in time, and seeking the interrelationship between those two only. In plotting the digraph, it shows the minimum number of relationships necessary to display the true picture.

3.4.2. POOLING OF KNOWLEDGE

Situations facing businesses today often need to be made following months of research into such areas as labour, raw materials, environmental impacts, market growth, economic growth, consumerism, competition, and improving technology - to mention but a handful. It is difficult enough just to have a good understanding of one of these areas, let alone understand several of them. This has led to a rise in teamwork in order to understand these issues better

ISM is an ideal team tool in that it allows for knowledge from people with different backgrounds and areas of expertise to be pooled. This is done because everyone has a say in the ISM process. This ensures that factors from all areas are included in the model, and that different perspectives are seen.

3.4.3. GROUP IDENTITY

Because ISM is a group process, when it is used in a management situation, the group of managers all feel part of the process, and thus part of the model. Thus, when policy is formulated from the results of the model, people can more easily identify with the decisions and policies. It is therefore a useful tool in giving direction to a company. Of course, to successfully do this, it is necessary for the 'stakeholders' or people most effected by decisions made from

the model to be those people involved in the modelling process.

3.4.4. REMOVAL OF HIDDEN AGENDAS

As has been mentioned, many answers are implied by the mathematics behind ISM. This factor, combined with the size of the matrix manipulations makes it impossible to keep track of the answers already given, those already implied by the process, and also the answers that will be implied by answering either 'yes' or 'no' to the pair currently under discussion.

This makes manipulation of the modelling process impossible, and rules out any power struggles or a scenario where the loudest mouth gets his own way. Instead, the group can only focus on the pair currently under discussion.

3.4.5. CROSS POLLINATION BETWEEN FUNCTIONS

Often in an ISM session there will be people from marketing, finance, engineering, personnel and other departments. These different functional heads will thus have a chance to meet and discuss issues far broader than there own departments. From this discussion, the individual participants are given insight into parts of business that are important to other departments that they have previously seen as unimportant or irrelevant. This cross-pollination between functions a produces better understanding between departments, and helps the company as a whole to function better.

3.4.6. CONTINUITY OF PURPOSE

In any company there are always people joining and leaving the company. In order for those joining the company to understand the company better and quicker, it can be advantageous to show them the results of an ISM done by the company. This helps

people to understand what the company's objectives are, as well as where they fit into the greater whole of the company.

3.5. THE LIMITATIONS OF ISM

Although ISM is a powerful a tool for grasping an understanding of complex situations, it is not without certain limitations :

3.5.1. NUMBER OF ELEMENTS

As has been mentioned, the time taken for an ISM session depends on the number of elements to be modelled. The number of possible questions is equal to n^2, where n is the number of elements. Thus increasing the number of elements from 30 to 37 will mean 52% more questions to be answered.

This limitation can be solved in part by choosing to limit the number of elements to twenty or thirty, and for the rest of the elements to be put in later by hand without the computer. This can further save time if only part of the group inserts these final elements into the model. This does, however, have its drawbacks.

For instance, in a priority structure, it is fairly easy to decide on whether one element is less important than another, especially as the digraph is linear (i.e. has many levels with only one element on each level). A linear digraph as shown in figure 3.3, ensures that only the element in the level below effect that level, and that that level effects only the level above. Thus a new element F, fits either before A, between A and B, between B and C, between C and D, between D and E or after E. However, only **one** of these six options is available.

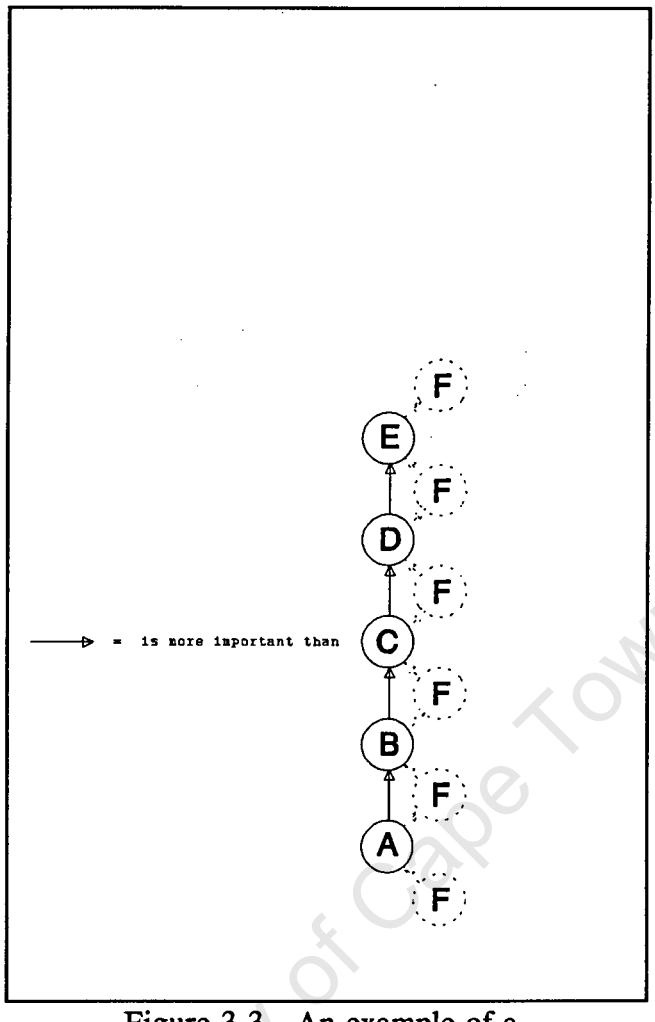


Figure 3.3 - An example of a priority structure

However, for an intent or attribute enhancement structure as is shown in figure 3.4, it is much more difficult to fit in an element, as elements from different levels may contribute to the new element, which in turn may contribute to elements at several levels. Thus each individual element has to be considered in a two-way relationship with the new element, a time consuming task. There are in fact $2n$ relationships that are all available (n being the number of elements prior to adding the new element), and so the placing of elements in this kind of structure is much more difficult.

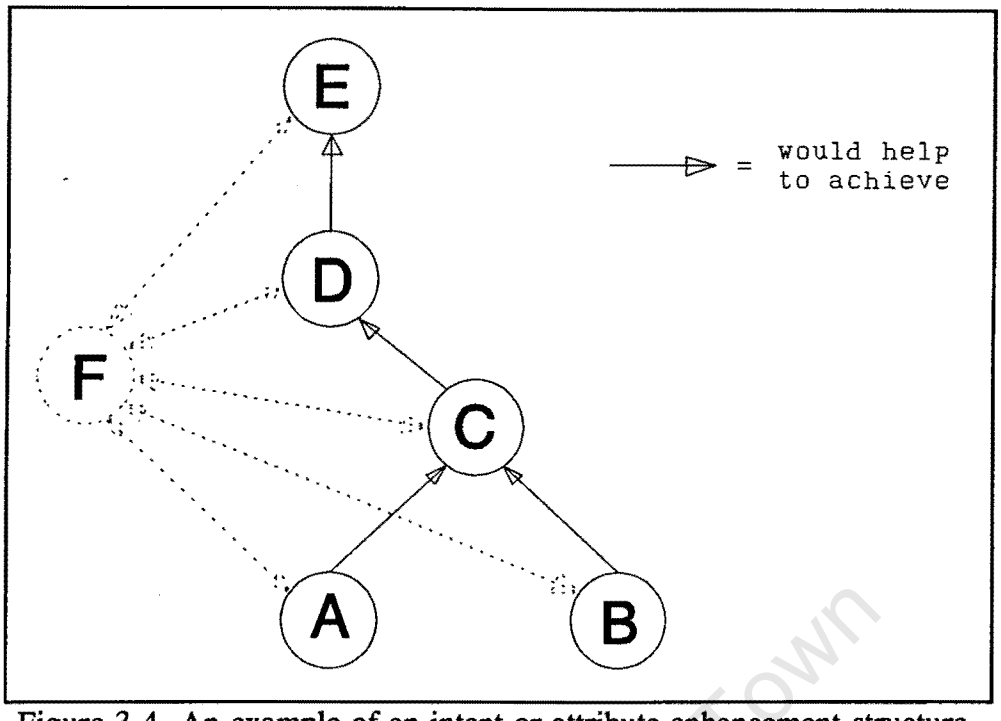


Figure 3.4. An example of an intent or attribute enhancement structure.

It is therefore often easier merely to limit the ISM to the chosen number of most important elements.

3.5.2. SIZE OF THE GROUP

Because of the discussion that occurs in an ISM session it is difficult to have more than eight people present. This is a problem if it is desirable for more people to be involved in the modelling process. As the group gets larger, not only does it take longer to discuss each question, but the quality of debate decreases as points are dropped and raised again as different group members speak. Not only does the modelling time increase, but the time taken in generating the element set also increases significantly.

3.5.3. REPETITION OF SOME ELEMENTS

Because of the fact that the ISM mathematics seek to imply the maximum number of answers (and ask the minimum number of questions) certain elements repeat themselves in the modelling process. (The reason for this is explained in Appendix A.) Because of this repetition of elements, people may feel that they are 'picking' on certain elements, while ignoring other elements. Also, people may complain that the pairs have to be considered both ways.

This can be alleviated by stressing at the start of the session the importance of the task being tackled, as well as explaining that the computer asks the questions that imply the most answers and that this can result in certain elements being seen very often, but that it does not effect the outcome in any way.

3.5.4. COMPUTER FACILITIES WITH SOFTWARE

As can be seen from appendix A, it is a momentous task to try to run an ISM session without access to a computer with the software. Fortunately, with the advances in computer technology, it is no longer necessary to have to use mainframe computers for ISM sessions. Today, even an IBM compatible with a 286 microprocessor is adequately powerful for a small ISM, (twenty elements or less), while the more powerful 486 microprocessors will process fifty elements with ease (more than enough for most situations.)

It is not easy to obtain ISM software. Various people have written ISM packages for the IBM PC, the Apple Macintosh, and various mainframe computers. For those people without access to the software, ISM is an impossible task. However, there is a listing of the ISM program developed as part of this thesis in appendix C.

3.5.5 CONTEXTUAL RELATION

Warfield (ref. 12, page 295) comments that the contextual relationship must have the property of transitivity. This is because the mathematics driving the process relies upon the transitivity of the relationship. A relationship is transitive if it has the following property : Should A relate to B, and B relate to C, if it can then be implied that A relates to C, the relationship is transitive. Warfield (ibid.) states that *While the relation "causes" is not necessarily transitive, the relation "impacts" would appear to be transitive in virtually all imaginable situations.... Preference is subjective, and subjective relations may or may not be transitive. Thus if a person says that 'blue is preferred to red' and 'red is preferred to yellow', it may still be that 'yellow is preferred to blue'; hence transitivity is violated.*

It is interesting to note Moore's (ref. 13, pages 82-83) comments on the matter : *The relationship is expressed in a subordinate phrase ... for example, they might say the 'Rather than deal with "Is item A LESS IMPORTANT THAN item B?" [a subordinate relation], we should use 'Is item A MORE IMPORTANT THAN item B?" [a superior relation]. As the underlying logic of ISM assumes a subordinate relation, it is critical that the subordinate phrase be used throughout the session.* Janes (ref. 20, page 148) disagrees with this in saying that *an associated contextual relation might be : 'Is more important than'; or 'Is better value for money than'.* Both of these are superior relations, and yet are cited as acceptable to by Janes.

In the light of Warfield's statements on transitivity, it is not necessary to have a subordinate relation. Transitivity is true both for : A>B, B>C, therefore A>C, as well as A<B, B<C, therefore A<C. Moore's statement is not true, in that it excludes many relationships that are transitive and do not violate the mathematical basis of ISM.

However adequate thought must be place into the choosing of the contextual relation to ensure that the mathematical theory of ISM is not violated.

Once one is aware of these limitations, the applications for ISM are limited only by the imagination.

4. THE ISM PROGRAM

Because ISM is still a developing methodology, and is not widely used in the business, ISM software is not freely available. ISM packages such as PRISM are expensive, and as the source code is not available, the package cannot be modified to suit a specific purpose. It was therefore decided to develop an ISM package, for the IBM PC which is the main contribution of this thesis. The package would then be available to the School of Engineering Management at the University of Cape Town for future research.

The development and structure of the ISM program is dealt with in detail in appendix B, and a complete listing of the source code is to be found in appendix C. In this section, only those aspects of the program seen by the user will be discussed.

The program has five main aspects, namely :

- Entering the elements, the context, and the contextual relation.
- Modelling the elements.
- Modifying the elements and the model.
- Plotting the digraph.
- Saving and restoring models from disk.

What follows is a description of each of these five aspects :

4.1. ENTERING THE ELEMENTS, CONTEXT AND CONTEXTUAL RELATION.

This option allows the modeller to enter the contextual relation, and the overall context in which the model is to be considered. The elements to be modelled are also entered at this point.

4.2. MODELLING THE ELEMENTS.

Once this option is selected, the whole modelling process must be completed in one sitting from start to finish. Unfortunately it is not possible to view the digraph as modelling progresses, or to save the model in an unfinished state. This is a limit of this particular program, rather than one of ISM itself.

4.3. MODIFYING THE ELEMENTS OR THE MODEL.

It may be that the modellers wish to modify, add or delete elements, or to modify the model itself. The program allows for elements to be deleted, for new elements to be added, or for the description of elements to be changed. Should new elements be added, the program can then be returned to the modelling option to model these elements into the existing model.

The model itself can be changed by cutting or removing links between elements, or by adding new links between elements. This process can also be used when adding new elements, but it is not recommended, as it will not guarantee that all possible relationships for that element have been determined.

4.4. PLOTTING THE DIGRAPH.

The digraph of the model is plotted on the screen with a selection of options on the bottom. The digraph is simple in that only the element number and not the full element description is displayed. The feedback sets are outlined by dashed lines, and solid lines are drawn to show the relationships between the elements. As digraphs can be plotted either by levels or by stages, both these options are available on a toggle basis. The display is very simple, and no attempts were made to try to ensure that lines do not cross, or do not lie on top of each other. However, by toggling between the plot by levels, and the plot by stages, it is possible to see the true relationships, and were lines do lie on top of each other

on the one graph.

Sometimes the digraph does not fit on the screen because it is too long or too wide. There are two options available when this happens. The digraph can be either be reduced in size by five percent at a time, or it can be compacted, which halves the size of the digraph in one step. If the graph is too small, it can also be enlarged.

As the digraph is plotted using element numbers and not the element descriptions, one can enter the element number to get a full description of it at the bottom of the screen. Only one element description can be shown at a time due to space constraints.

The option of printing the digraph is currently not available. This is due to the fact that *Pascal* offers no advice on how to print graphics screens. It is possible to print the screen using the *Print Screen* command on the keyboard, but this can only be done with certain versions of *DOS*.

4.5. SAVING AND RESTORING MODEL FROM DISK.

One can save the elements, context, contextual relation and the matrix of the model onto a disk. This model can then be retrieved later on when another look at it is desired.

This is a very useful function as the contextual relation, context and elements can be saved in an unmodelled (or blank) state. When the modelling is about to start, these can be restored from disk. This function was used for all interviews, where a 'blank' model was stored on disk, and retrieved at the start of the interview.

It also opens up the possibility of doing a 'postal' ISM, where each person could be posted a disk with the software already on it. All the respondent has to do is put the disk in the

drive, and turn on the computer, and the package boots, loads the blank model and starts asking the questions. When modelling is complete, the program saves the model, and the disk is posted back to the interviewer. (Of course, the respondents need to be familiar with ISM....)

Also, the model is freely available to be looked at later, and to be compared by computer to another model. This is useful, as an ISM model contains far more information than it appears to at first glance.

From this one can see that the package is indeed comprehensive, and covers all aspects desired by a user except for the printing of the digraph.

5. DETAILS OF THE MODELLING PROCESS

This section describes what happens in the matrix of the model as the questions asked by the computer are answered. It does not attempt to explain why certain answers are implied or the underlying mathematics of ISM. These two points are dealt with in appendix A. This section will, none-the-less, give a basic insight into the process of modelling.

For this example, a six element ISM will being modelled. The contextual relation is R, and the elements are A,B,C,D,E and F. The following modelling process will take place :

The element A will be selected, and the computer will ask the questions 'does A relate B', 'does A relate C', does 'A relate D', 'does A relate E' and 'does A relate F'? Assuming that the answers 'yes', 'no', 'yes', 'no' and 'no' are given to those questions, the matrix will then be arranged as follows :

	B	D	A	C	E	F
B	1					
D		1				
A	1	1	1	0	0	0
C				1		
E					1	
F						1

(5.1.)

The element A remains as the central element, and the questions 'does B relate A', 'does D relate A', 'does C relate A', 'does E relate A' and 'does F relate A' are then asked in that order. Assuming that the answers 'yes', 'no', 'yes', 'no' and 'no' are given, the matrix will be ordered as follows :

	D	B	A	E	F	C
D	1		0			
B		1	1			
A	1	1	1	0	0	0
E			0	1		
F			0		1	
C			1			1

(5.2.)

From the answers given, answers will be implied, and the matrix will look as follows :

	D	B	A	E	F	C
D	1	0	0	0	0	0
B	1	1	1	0	0	0
A	1	1	1	0	0	0
E		0	0	1		0
F		0	0		1	0
C	1	1	1			1

(5.3.)

The modelling process will then focus on elements E and F, and the questions 'does E relate F' and 'does F relate E' will then be asked. Assuming the answers 'no' and 'yes' are given, matrix 5.4. will result.

	D	B	A	E	F	C
D	1	0	0	0	0	0
B	1	1	1	0	0	0
A	1	1	1	0	0	0
E		0	0	1	0	0
F		0	0	1	1	0
C	1	1	1			1

(5.4.)

The process will now seek to fill in the missing blanks in the matrix. This is done by forming an implication matrix, that allows the process to select the question that when answered will result in the most implied answers.

The implication matrix will result in the question 'does F relate D' being asked, and the relevant answer will be inserted into the matrix. Assuming that the answer 'no' is given, then the answer to the question 'does E relate D' is implied to be 'no' as well.

Another implication matrix will be formed, and the question 'does C relate F' will be asked. Assuming this is answered 'yes', then the answer to 'does C relate E' is implied to be 'yes'. This makes the modelling process complete, and the matrix will be as follows :

	D	B	A	E	F	C
D	1	0	0	0	0	0
B	1	1	1	0	0	0
A	1	1	1	0	0	0
E	0	0	0	1	0	0
F	0	0	0	1	1	0
C	1	1	1	1	1	1

(5.5.)

From this the digraph shown in figure 5.1 can be obtained.

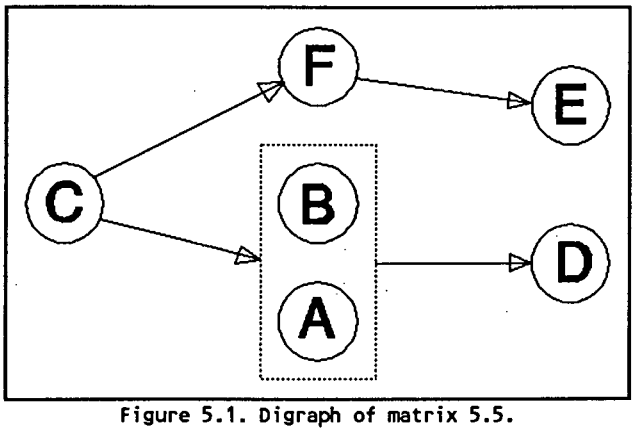


Figure 5.1. Digraph of matrix 5.5.

PART II

TOTAL QUALITY MANAGEMENT

6. TOTAL QUALITY MANAGEMENT

Total Quality Management (TQM) has become a 'buzz-word' in industry today. TQM may be defined as a management method that 'involves the co-operation of everyone in an organisation and associated business process to furnish products and services which meet the needs and expectations of customers.'[1]

TQM cannot be said to be the work of any one individual. Indeed, several people have produced books on quality management that meet this definition. TQM was developed in Japan through the work of Deming, Ishikawa and Juran, whose ideas have only recently begun to be practised on a global scale. In America, Crosby has been better known for his fourteen points of quality. TQM is in effect a synthesis of the work of these and other authors.

This section looks at the works of Deming and Crosby, followed by a synthesis of how these and other authors have been synthesized to form TQM. Deming's work was chosen because of the major influence it has had on Japanese industry. Crosby was chosen as his ideas on quality management are those most often encountered by South African managers. Also, Crosby has laid out his 'plan of action' in introducing quality management far better than any of the other authors. The work's of both author's are particularly relevant to South Africa, in that they deal with management attitudes toward the workforce, and place management of people above the need for technology, an area where the other author's are not as strong.

6.1 THE WORK OF W. EDWARDS DEMING

Deming originally started working with statistical sampling techniques, focusing on variability in manufacture. His early work was based on the theories proposed by Dr. Walter Shewhart[35] while he was working at Western Electric. Shewhart differentiated between 'special' and 'common' causes of variability, and argued that if the cause of variability could

be located, it could then be removed. Once the variability was removed the product would be improved. Shewhart developed statistical process control charts in order to put his theories into practice, and Deming followed in his footsteps.

Shewhart proposed monitoring a process, examining the results, and then acting on them to improve future results.Deming developed this into a four step cycle of *plan*, *do*, *check* and carry out *action*, known as the PDCA cycle. His early work in quality management earned him many awards in Japan. However, his ideas met strong resistance in North America.

More recently, he has arrived at 'five deadly diseases' that are plaguing quality in industry, namely :

1. A general lack of constancy of purpose to plan a product or service that will have a market and keep the company in business.
2. Too much emphasis on short-term profits.
3. Evaluation of performance, merit rating or annual review.
4. Management is too mobile - job hopping
5 Management decision-making by visible figures, without paying due consideration to less tangible or hidden factors.

It be said that a company suffering from these diseases is in the middle of a messy problem and is not even aware of it. The five deadly disease are unbounded (all companies suffer from this problem, and it is culturally acceptable), a large number of people are involved(all employees and future employees are involved), the solution to the problem lies in the long-term and not in the short-term, and the solution is unknown or it would have already been implemented. Thus Deming is trying in his approach to quality management to deal with a messy problem.

Deming proposes his fourteen points which he believes are the 'anti-dote' to the diseases (or the solution to the mess) :

1. Create constancy of purpose to improve product and service, with the aim of becoming more competitive.
2. Adopt a new philosophy for the new economic age with management learning what their responsibilities are, and by assuming leadership for change.
3. Cease dependence on inspection to achieve quality by building quality into the product.
4. End awarding business on price. Award business on total cost and move toward single suppliers and long-term loyalty.
5. Improve constantly and forever the system of production and service to improve productivity and quality and to reduce costs.
6. Institute training on the job.
7. Institute leadership with the aim of helping people to do a better job.
8. Drive out fear so that everyone can work effectively.
9. Break down barriers between departments. Encourage research, design, sales and production to work together to foresee difficulties in production and use of the product or service.
10. Eliminate slogans, exhortations and numerical targets for the workforce. They are divisory, and the difficulties belong to the whole system, not to the workers.
11. Eliminate quotas or work standards, and management by objectives or numerical goals; leadership as in 7 above should be instituted instead.
12. Remove barriers that rob people of their right to pride in their work. Supervisors should worry about quality, not volume.
13. Institute a vigorous education and self-improvement program.
14. Put everyone in the company to work to accomplish the transformation.

In order to achieve these points, the following action plan is recommended to bring about the necessary culture change within the organisation :

1. Agree to adopt the new philosophy.
2. Have the courage to break from tradition, and take pride in the new philosophy and responsibilities.
3. Explain to everyone why change is necessary, and that everyone will be involved.
4. Divide the work into stages, each proceeding stage is a customer, and must be accommodated as such.
5. Construct an organisation to guide continual improvement of quality.
6. Everyone is to aim at improving the inputs and outputs of any stage, with ideas, plans and figures.
7. Embark on construction of the organisation for quality.

Deming's work is the result of 50 years in industry, and is not to be taken lightly, especially in the light of what is happening in Japan. His point on creating constancy of purpose is possibly his most important as when production is needed urgently, all the objectives and plans of quality in manufacture tend to be the first to be left behind.

Another valid point he raises is that of working closely with one's suppliers. It stands to reason that the quality of what is sent out of a plant cannot exceed the quality of the raw material brought in. Thus in the long-term, one has to ensure that one's suppliers are also constantly aiming at higher levels of quality.

His ideas of eliminating quota's and slogans and revolutionary to Western management. It almost seems like the ideas of a madman. However, he argues very convincingly that the real problems in companies are not caused by the production worker, but by the managers. It is quite clear from his work that the managers are firstly and foremostly responsible for quality. Such things as the worker training, ensuring that proper tools

are available, and the raw material purchased are not the responsibility of the worker. Why then should the worker carry the can when it is not their fault? This is indeed tough medicine, and quite possibly why his work is not catching on in the Western world.

However, his works do not offer practical answers on how to better structure a company for achieving quality. He states what needs to happen, but offers no definitive plan on how best to convert the plan to action. He leaves this very much to the reader to come up with his own answers for his own company. While this is a definite drawback to a casual reader, his work is based on broad principles, and perhaps asking for any form of company structure would be too much. However, a person who seriously plans to act upon his writings, would with some effort arrive at a structure that would best suit his needs.

6.2. THE WORKS OF PHILIP B. CROSBY

Crosby's quality philosophy is rooted on his four absolutes of quality, namely :

1. Quality has to be defined as conformance to requirements, not as goodness.
2. The system for causing quality is prevention, not appraisal.
3. The performance standard is 'Zero Defects'.
4. The measurement of quality is the price of non-conformance, not indices.

Crosby, unlike Deming, uses slogans such as 'do it right the first time', and 'conformance to requirements' to promote quality. He believes that there are no 'economics of quality' and that it is always cheaper to do it right the first time. He argues that errors are not part of everyday life (no-one would ever go home to the wrong house), and so they should not occur in business. Either the product conforms to requirements and is thus a quality product, or it does not conform and is

a non-quality product. Quality must be the company's number one goal.

Crosby also believes that there is no such thing as a 'quality problem.' 'Problems' are created by the management process, and today's problems come from yesterday's solutions. Management is there to provide an environment in which the ongoing process of quality improvement can be fostered. The primary aim of managers is to improve the attitude and activities of the workers.

He has developed a fourteen step process of quality improvement :

1. Establish management commitment - each person must be personally committed to the program in order to raise the visibility of quality.
2. Form a quality improvement team to run the quality improvement program. Each department must have a representative on the team.
3. Establish methods of measuring each activity in the company.
4. Evaluate the cost of quality by adding up the cost of scrap, rework, etc. This will aid in getting commitment to the quality program.
5. Make people aware of quality via posters, booklets, videos, and by training supervisors.
6. Take corrective action - i.e. eliminate the source of the problem forever. Make sure the person who has the authority to solve the problem is aware of the problem.
7. Plan the official launch of the 'Zero Defects' (ZD) program, and how best this program can be implemented.
8. Educate the supervisors and workers about the quality improvement program so that it can be properly implemented.
9. Hold ZD day so that all employees can sign their commitment to improvement.
10. Get employees to set their own goals, and have them prominently displayed.

11. Set up an error cause removal program to enable problems that prevent perfect work to be identified and removed.
12. Give recognition to outstanding performers by (non-financial) awards.
13. Establish regular meetings of the quality professionals (Quality Councils) to discuss problems, solutions and experiences.
14. Do the whole process over again.

Crosby believes that a company can be 'vaccinated' against non-conformance. This is done by building anti-bodies into the management style of the company. Crosby recommends the following vaccine :

1. Integrity - the whole company is committed to giving the customer what was promised.
2. Systems - systems for quality control are in place to educate, report deviations, measure cost of non-conformance etc.
3. Communications - quick, continuous information flow on waste, opportunities, and achievements.
4. Operations - educate and support suppliers, modify procedures, products and systems as needed, and keep training.
5. Policies - clear and unambiguous, with the quality function reporting at the same level as the functions it measures.

Crosby has also drawn up a 'Quality Management Maturity Grid', designed to help measure a company's progress in quality. On one axis are the five stages of maturity, and the other axis has six management categories which each have to be rated.

The five stages of maturity are :

1. Uncertainty - no commitment to quality, and problems go unsolved.
2. Awakening - volume still number one issue, only short-term solutions are sought, and no resources are committed to permanent improvement.

3. Enlightenment - quality department is established, and problem solving begins in earnest.
4. Wisdom - Everything is running smoothly, permanent quality improvements are made. Cost reductions are taking place, and problems are being removed.
5. Certainty - It is well known why there are no quality problems.

On the other axis are the six management categories that are assessed :

1. Management understanding and attitude - from no understanding to quality being an essential part of the company.
2. Quality department's organisation status - from inspection in manufacturing to represented on the board of directors.
3. Problem handling - from fire fighting to problem prevention.
4. Cost of quality as a percent of sales - from an unknown amount in the region of 20% down to a known and definite 2.5%.
5. Quality improvement actions - from all talk and no action to a continual normal activity.
6. Summation of company quality posture - from not knowing why there are quality problems to knowing why there are no quality problems.

Crosby's work is filled with examples of successful applications of his fourteen points, almost to the point at which it is made to look to easy. In fact, after reading his books, one may well believe that there truly is no such thing as a 'quality problem', and that it will merely take a few short weeks to sort out the problems. However, he does make it clear that it will, in fact, take several years to really sort out problems.

His quality management maturity grid is a very useful tool for measuring achievement when implementing a quality program. He wisely cautions that just because a company has moved up one

step on the grid does not mean that it will not retreat at a later date, and that continuous improvement is needed.

His work is much clearer than Deming's when it comes to how a company should be structured to implement a quality program. This is particularly useful, as he states that quality control should not report to manufacturing, but to the same level that manufacturing reports. He states clearly that everyone is responsible for the quality of their own work, and that the quality department is there to assist people in achieving zero defects, and not to inspect the work.

Like Deming, he also advocates training of the workforce, and also advocates working more closely with suppliers. Crosby does put more emphasis on the fact that quality is not merely something for a factory, but that it should be applied to the service industry too.

6.3. THE MAIN PRINCIPLES OF TOTAL QUALITY MANAGEMENT.

From the work of Deming, Juran, Crosby, Ishikawa and others, the management philosophy of TQM has been born. TQM is not a 'program', but a process by which a company ensures that its service or goods meet the customers expectations. TQM is best described as a long-term commitment by a company, to change the culture of the organisation so as to mobilise all employees to work together for the purpose of improving the quality of work in every area of the company.

TQM cuts across the grain of the ideas of inspecting products and rejecting the defects. Rather, TQM aims to remove the causes of defects by understanding why they occur. To do this, all aspects of the company from design through to purchasing and from production through to accounting all need to work together to eliminate defects.

A very important aspect of TQM is that unlike traditional

quality initiatives that dealt only with production lines, TQM also aims at the service industry and the office environment, advocating a reduction in errors by improving quality of work. Most authors do concentrate on the production line, Crosby is a firm believer in effecting change in the office.

There are common principles that run through the works of these authors upon which TQM has been founded. For further reference, table 6.1. shows how Juran differentiates between 'big Q and little Q.' Juran advocates that 'big Q' is the correct method.

The key points of TQM are as follows :

COMMITMENT

TQM relies on the commitment to quality that springs first and fore-most from top management and the chief executive officer. From the top it flows down to result in a commitment to quality by the middle and lower managers, and then the workforce. The commitment of the top manager will result in the status of quality in the organisation rising to be of primary importance, which will in turn result in a 'culture change' within the organisation. (Deming's 1st point, Crosby's 1st point, and Juran's comments that quality is to be co-ordinated by a quality council of upper mangers.)

EDUCATION AND TRAINING

All levels of the company need to be educated as to the true cost of quality in the company. I.e. the cost of mistakes, rework, warranties etc. Only once people are aware of the magnitude of the saving possible should training begin. Training must teach people what quality means, what is expected of them, how they are meant to do their job, and what to do when problems occur. (Deming's 13 point, Crosby's 8th point, and Juran's notes that training in managing quality is company-wide.)

Topic	Contents of little Q	Contents of big Q
Products	Manufactured goods	All products goods and services, whether for sale or not
Processes	Processes directly related to manufacture of goods	All processes; manufacturing support; business, etc.
Industries	Manufacturing	All industries; manufacturing; service; government, etc. whether for profit or not
Quality is viewed as :	A technological problem	A business problem
Customer	Clients who buy the products	All who are impacted by, external and internal
How to think about quality	Based on culture of functional departments	Based on the Universal Trilogy
Quality goals are included :	Among factory goals	In company business plan
Cost of poor quality	Costs associated with deficient manufactured goods	All costs which would disappear if everything were perfect
Improvement is directed at :	Departmental performance	Company performance
Evaluation of quality is based mainly on :	Conformance to factory specifications, procedures, standards	Responsiveness to customer needs
Training in managing for quality is :	Concentrated in the quality department	Company-wide
Co-ordination is by :	The quality manager	A quality council of upper managers

Table 6.1. Contrast big Q and little Q (adapted from ref 6, page 12.)

CONTINUOUS IMPROVEMENT

TQM proposes continuous improvement. It is not good enough to improve and to stop improving. In order to improve, it is necessary to ensure that there are proper methods of measuring quality. Without measurement it is impossible to know if improvements have been made.

Measurement should pinpoint where defects are occurring, and show wether the problem is one of training, purchasing or limitations of the machinery. This will enable fast and effective action to be taken when dealing with problems.

Also, people should be encouraged to give input on how to solve quality problems, and to offer ideas on how to improve the plant as a whole. (Deming's 1st, 5th, and 14th points. Crosby's 3rd, 6th and 11th points. Juran's statement on improvement being directed at company performance.)

TEAMWORK

Teamwork is encouraged, and the different functions should confer with each other to result in improvements. Thus, designers should not design without first inquiring with manufacturing as to how the parts are to be made. This helps to match design to processes, and helps to reduce errors. (Deming's 9th and 14th points. Crosby's 2nd and 13th points. Juran includes all people in his training on quality management.)

CUSTOMER-SUPPLIER RELATIONSHIP

The company should improve its relationships with its suppliers. This is to ensure the quality of goods being received is safe-guarded against sudden drops is quality. Also, it allows for processing or assembly problems with raw materials to be taken up with the suppliers to help reduce

errors.

Also, each person in the company is both a customer to the previous process, as well as a supplier to the next process. Each person must ensure quality work is given to their customers, and that they received quality work from their suppliers. (Deming's 4th point. Crosby (ref. 3, page 9 states that 'suppliers are educated and supported in order to ensure that they will deliver services and products that are dependable and on time.' Juran (ref. 6, page 359) recommends the auditing and rating of suppliers.)

PLANNING AND GOAL-SETTING

TQM aims to improve quality by setting goals for better production and planning to achieve those goals. Quality does not just happen, it has to be planned for. Juran is very strong on this point, and rightly points out that people plan financially with budgets, for production with schedules, develop marketing strategies etc. etc., and yet fail to plan to produce better quality. He argues that this is the reason for poor quality. (Deming's plan, do, action, check cycle. Crosby's 10th point.)

PART III

THE APPLICATION OF ISM AND TQM

7. APPLICATION

Once the program had been completed, and testing had shown it to be working properly, a series of models were made on the subject of TQM. These models were made by people employed within the South African clothing industry.

7.1. OBJECTIVE

The objective of the research was uncover people's perceptions of the relationships between important principles within a quality management initiative in the clothing industry.

A secondary objective was to see whether ISM is a methodology that can be used for interviewing individuals, and from the interviews gathering enough data to synthesize a meaningful model.

It was hoped that this would allow one to establish whether there is a dominant mental model of how aspects of quality management inter-relate. From this it was hoped that inferences could be made as to how best to implement TQM within a clothing company.

It was decided that individuals should be used to produce a series of models, rather than producing models from a series of groups. This was done for the following reasons :

1. It was hoped that by comparing different models it would be possible to establish a dominant model among the people interviewed. If this was done in groups, less models would be produced for comparison.

2. It is difficult to gather groups from industry together to discuss issues for research purposes. People from different companies often do not wish to share their expertise and knowledge with others from other companies - something that would be essential to produce a meaningful model. In order to

circumvent this, each group that produced a model would have to come from one company, which would be a very difficult to do.

3. It is easier and more flexible to do go to a person in their office at their convenience, than to try to gather groups together.

7.2. THE SOUTH AFRICAN CLOTHING INDUSTRY

The South African clothing industry is a major employer in the country. It has been an industry highly protected by tariffs. However, following the GATT agreement and the new government in the country, the minister of Trade and Industry, Trevor Manuel, has announced a radical reduction in import duty on clothing and textiles.

Even before this reduction, the economic depression in South Africa, and the cheap manufacturing of clothes in the Far East were placing pressure on the industry, and had resulted in a gradual decline in the number of people employed in the industry.

In order to counter this, companies are looking toward TQM as a method of remaining competitive by reducing internal costs, and improving quality and thus consumer loyalty. It was therefore seen as a good industry in which to produce models on quality management.

7.3. THE TYPE OF ISM PRODUCED

As was mentioned in section 3.2., there are three main types of ISM - priority structures, intent structures and attribute enhancement structures. Due to time constraints on the people interviewed, only one model could be made per person. It was therefore necessary to consider what type of ISM structure was best suited to fulfil the objective of the research.

Introducing TQM requires a change in management style, or a culture change within an organisation. This change results in some friction, and a certain amount of patience is called for while sorting out problems. The research was aimed at trying to see the inter-relationships between factors or aspects of TQM, so that one can see what effects change in one area will have in another. This would also allow for one to see where one should begin in implementing a TQM initiative to try to ensure both a smooth and a quick transition.

In order to achieve the desired result, a priority structure would be of no use, as it only ranks elements in order of importance. Importance does not help in finding root causes. It is important to pay one's debts, but the start point in paying is earning the money. Also, as TQM is a management philosophy aimed at ensuring continual improvement, and not at the reaching of a series of objectives, an intent structure is not well suited to TQM. Only an attribute enhancement structure was seen to be able to uncover the desired effect of inter-relationship between factors.

So, an attribute enhancement structure was chosen. The contextual relation of 'improving' was chosen, as TQM has the underlying philosophy of improving processes, systems, methods, training etc. All questions were asked in the following format : 'In the context of producing high quality garments would improving factor A improve factor B.' Thus it was hoped that the models obtained would help to show how improvement in quality is brought about. From the models, it was hoped that the lower level elements would be identified that would be points of leverage in initiating quality management. By starting on those base factors, a trickle-down effect would hopefully ensure the fastest possible implementation of the higher levels.

7.4. THE ELEMENT SET

It was decided that each person interviewed should use the same

element set. This was necessary as if each person had make their own element set, some may have made five elements, while other may have made fifty. This would make the comparison of models an impossible task, and a common element set would circumvent this. Also a pre-determined element set would greatly reduce the time for making each individual model.

Due to time constraints on people in the business world who were to create the models, it was also decided to limit the element set to fifteen elements. It was hoped that this would reduce the time for modelling itself to less than half-an-hour, making the whole interview at most an hour long, and yet still have sufficient elements to obtain meaningful data.

A set of elements was established by drawing upon the works of Deming, Crosby, Juran and others. A draft list was made first, by listing aspects of quality management that were common to two or more authors. Also included in the list were those aspects of quality management that a particular author viewed as important in managing quality, even if no other author placed emphasis on them. This list was shortened to fifteen factors by choosing those deemed to be the most important in the opinion of the various authors, or which were viewed as being particularly relevant to the clothing industry in South Africa. Every attempt was made to ensure that the elements were concise and that they were clearly stated to prevent misunderstanding.

Following a set of trial interviews with four people, some of the elements were modified to enhance clarity. The final element set is shown in table 7.1.

1. Status of quality within the organisation.
2. Quality of raw material purchased from the suppliers.
3. Methods of manufacturing garments (i.e. processes, machinery and equipment.
4. Commitment of the workers to quality.
5. Commitment of the managers to quality.
6. Awareness of the effect of quality manufacturing on profits.
7. Feedback on the quality of the work currently occurring on the shop floor.
8. Level of resources available on the shop floor to ensure the quality of garments during manufacture.
9. Quality improvement programs running on the shop floor.
10. Inherent quality designed into the garment.
11. Methods for prevention and removal of quality problems.
12. Education and training of managers in quality control.
13. Education and training of workers in quality control.
14. Recognition programs for people who achieve outstanding quality in their work.
15. Ability to pinpoint defects on the shop floor.

Table 7.1. - Element used for creating the models.

7.5. THE METHOD OF GENERATING THE MODELS

The modelling was done using the package developed as part of this thesis. However, before the model could be made, it was essential that the modeller was aware of what ISM was about. For this reason an explanation of ISM was given to each person along the following lines :

A brief explanation was given of what ISM stands for, and what it is about. This included such aspects as the fact that it was a process by which relationships between factors were found by asking questions. The different factors were paired

together, and the modeller was to answer whether the two did or did not have the given relationship between them. All questions were answered either 'yes' or 'no'. It was explained that the program used maths to imply answers, and so not all possible combinations of pairs would be asked.

An explanation was given as to what output was given by the package, and how this could potentially be used by the modeller to his or her benefit. It was also explained that there were no right or wrong answers, as the output reflected their interpretation of the relationships, and that this would probably be unique. The modellers were then given an opportunity to ask any questions that they might have had.

Once an explanation of the process had been given, as well as an example of how each question would be asked, the modeller was given a list of the elements to read in order to familiarise himself/herself with the elements. Before modelling began, the questions shown in table 7.2. were asked of the modeller in order to assess his/her experience and knowledge.

1. Have you done a formal tertiary course in quality control?
2. Have you done an in-house course/seminar/training session in quality control?
3. Have you read any books by Philip Crosby, Joseph Juran, W. Edwards Deming or Kauoru Ishikawa?
4. Do you actively support your company's quality program?
5. How do you rate your quality program (1 = poor, 10 = excellent)?
6. Approximately how many books/articles on quality control have you read in the last two years?

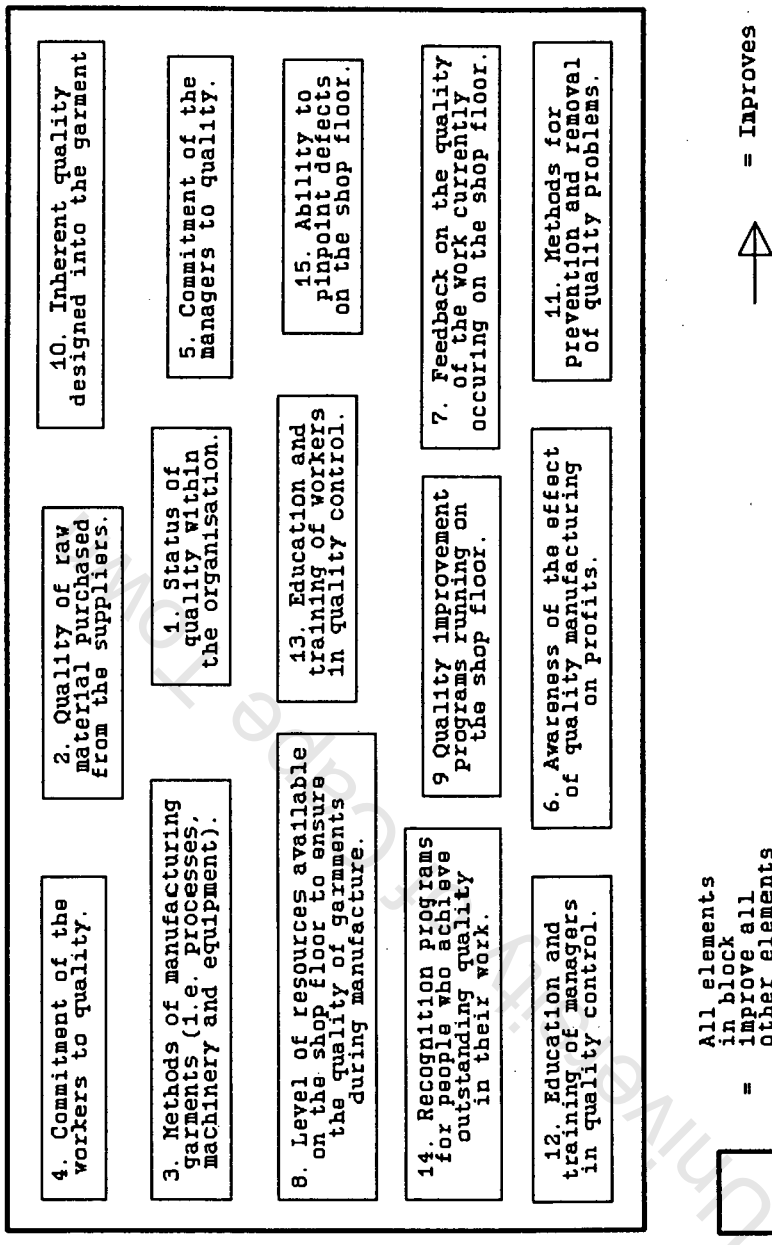
Table 7.2. List of background questions.

The people who created the models were chosen on the basis of their experience in the clothing industry, and where possible, people who were actively involved in quality programs.

Modelling then began, and where possible, the modellers typed in the answers to the questions themselves.

Once modelling was completed, the model was saved, and then displayed on the screen for the modeller to see. Because of the display does not give a full element description, these were done manually on computer, and sent to the modellers, together with a letter thanking them for their time. A total of nine interviews were done.

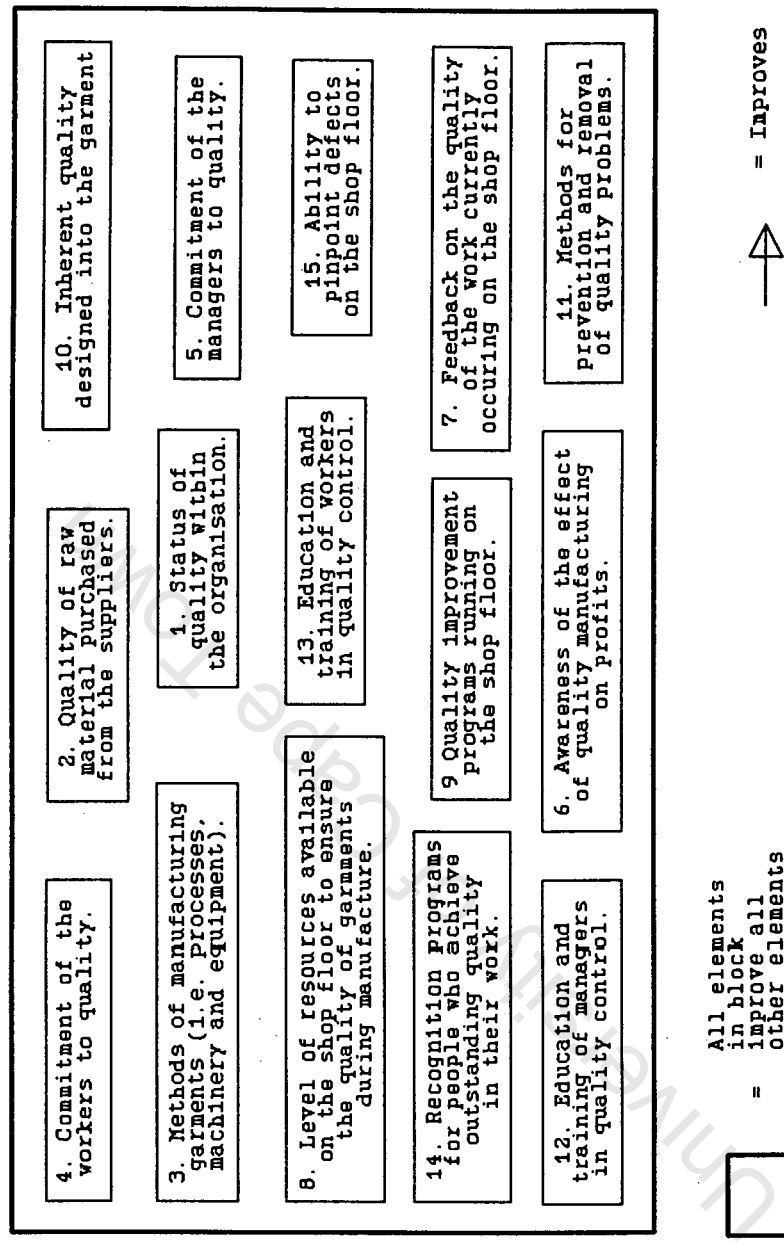
Once all the individual interviews were complete, a group ISM was done by four people, each one familiar with both TQM and the textile or clothing industry. For this model, an attempt was made to complete it using the thoughts and ideas of Deming, especially his five deadly diseases and his fourteen points. This was done in order to see whether the results obtained from the various modellers in any way correlated with the writings and thoughts of Deming, and whether his work was applied in the clothing industry.



Model 1

Question and answers

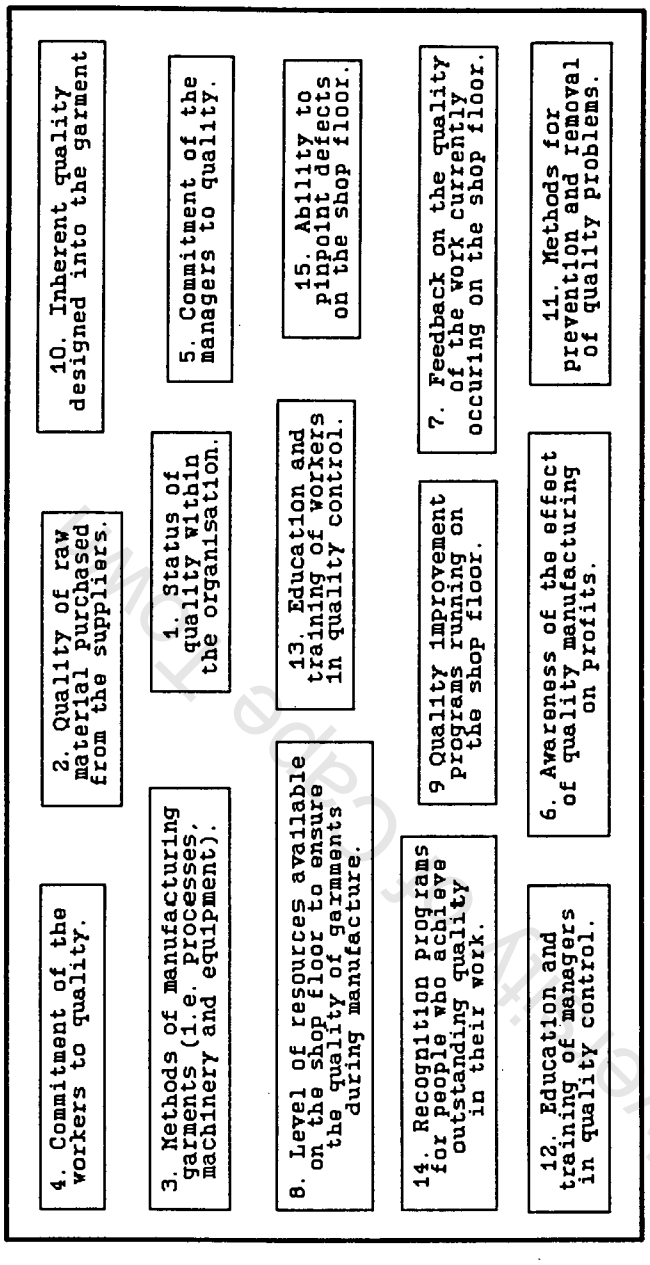
1. No
2. No
3. No
4. No
5. Not applicable
6. 0
7. 4



Question and answers

1. No
2. Yes
3. No
4. Yes
5. 9
6. 20
7. 0

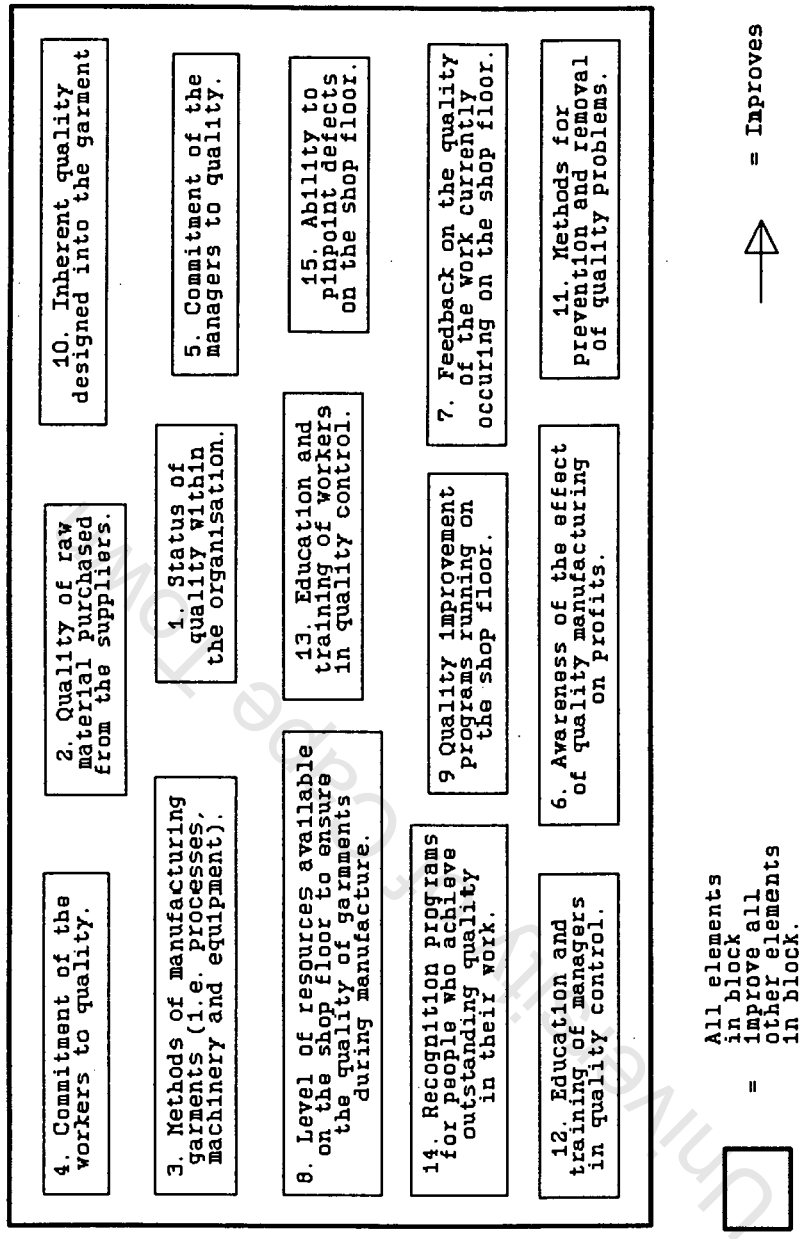
Model 2



Model 3

Question and answers

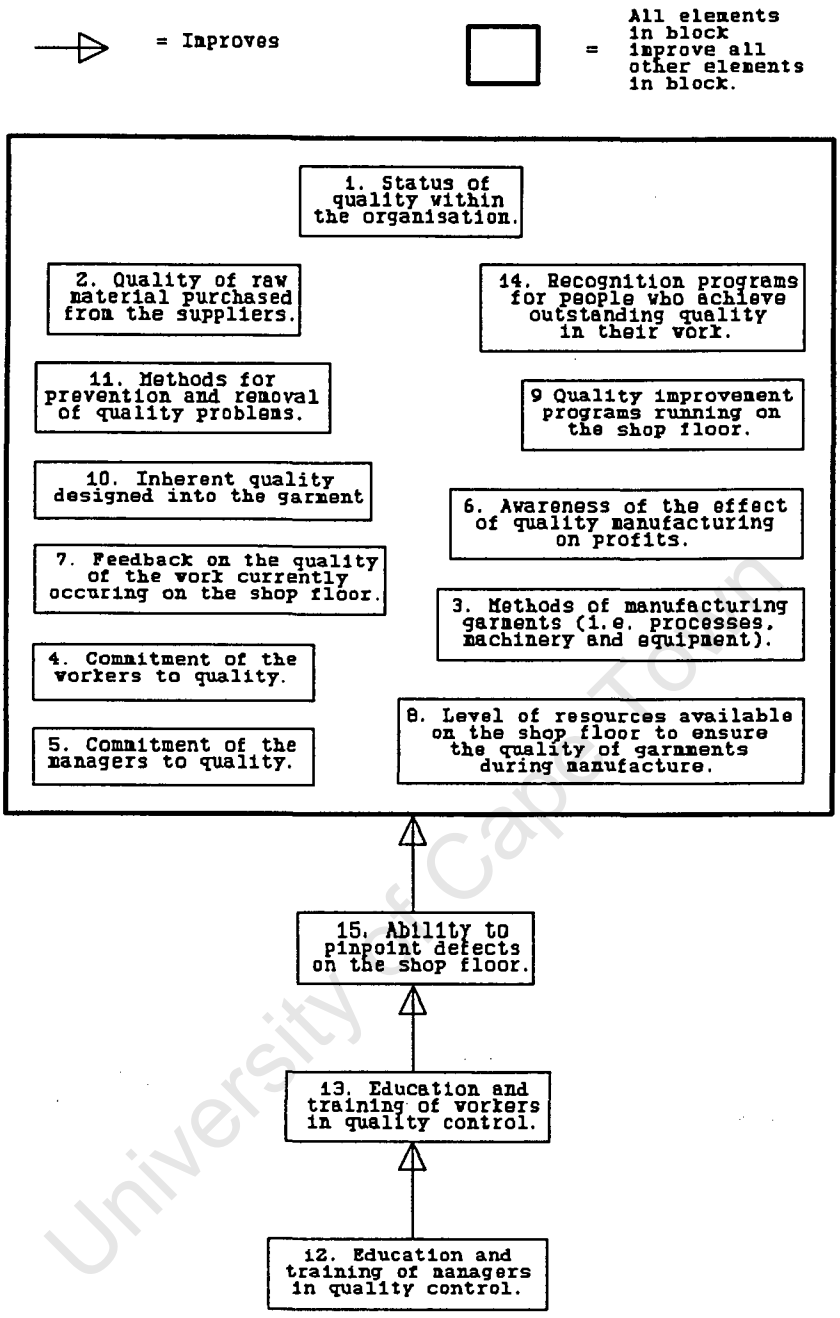
1. Yes
2. Yes
3. Yes
4. Yes
5. 8
6. 10
7. 10



Model 4

Question and answers

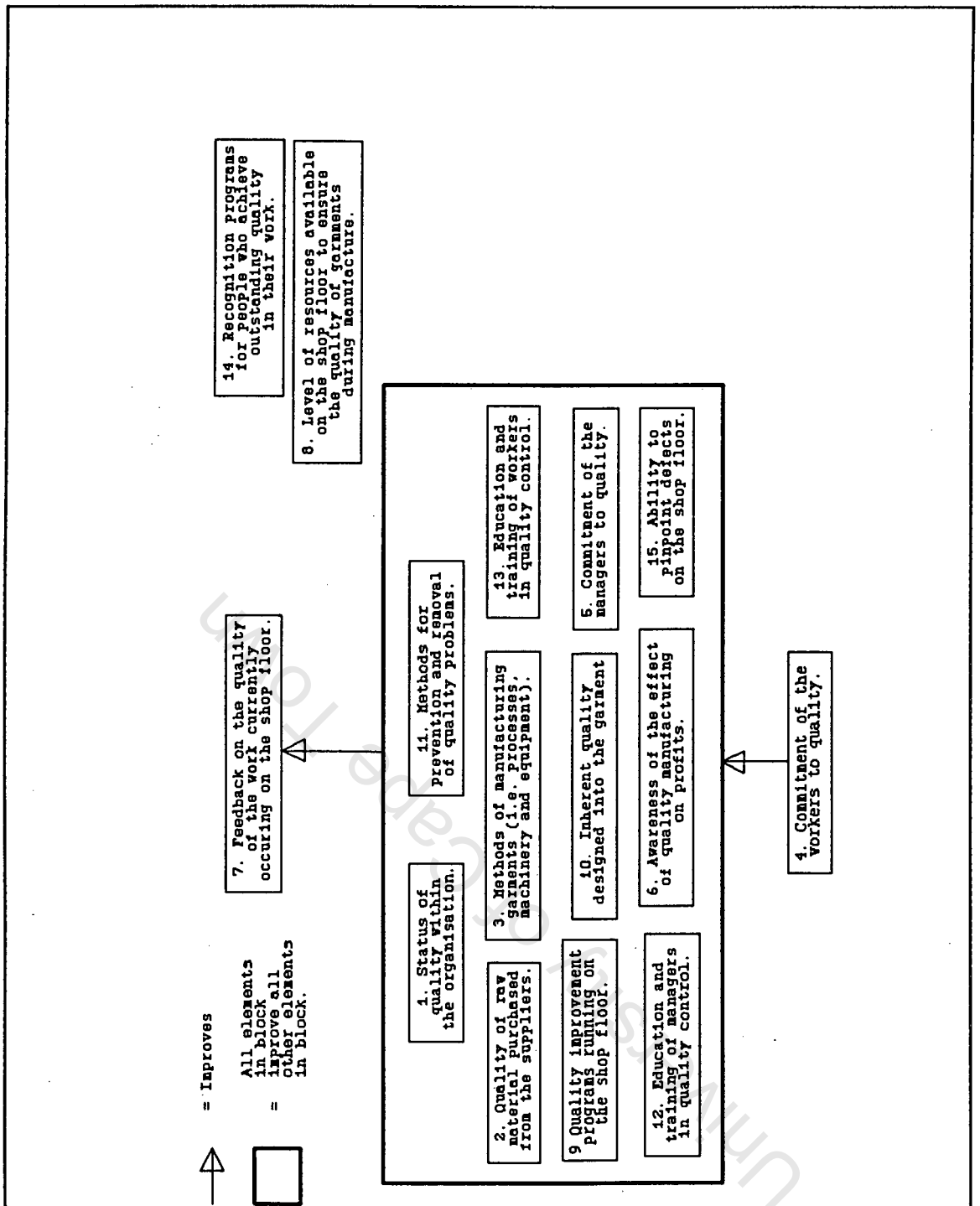
1. No
2. No
3. No
4. No
5. 7
6. 10
7. 4



Question and answers

1. Yes
2. Yes
3. Yes
4. Yes
5. 8
6. 7
7. 20

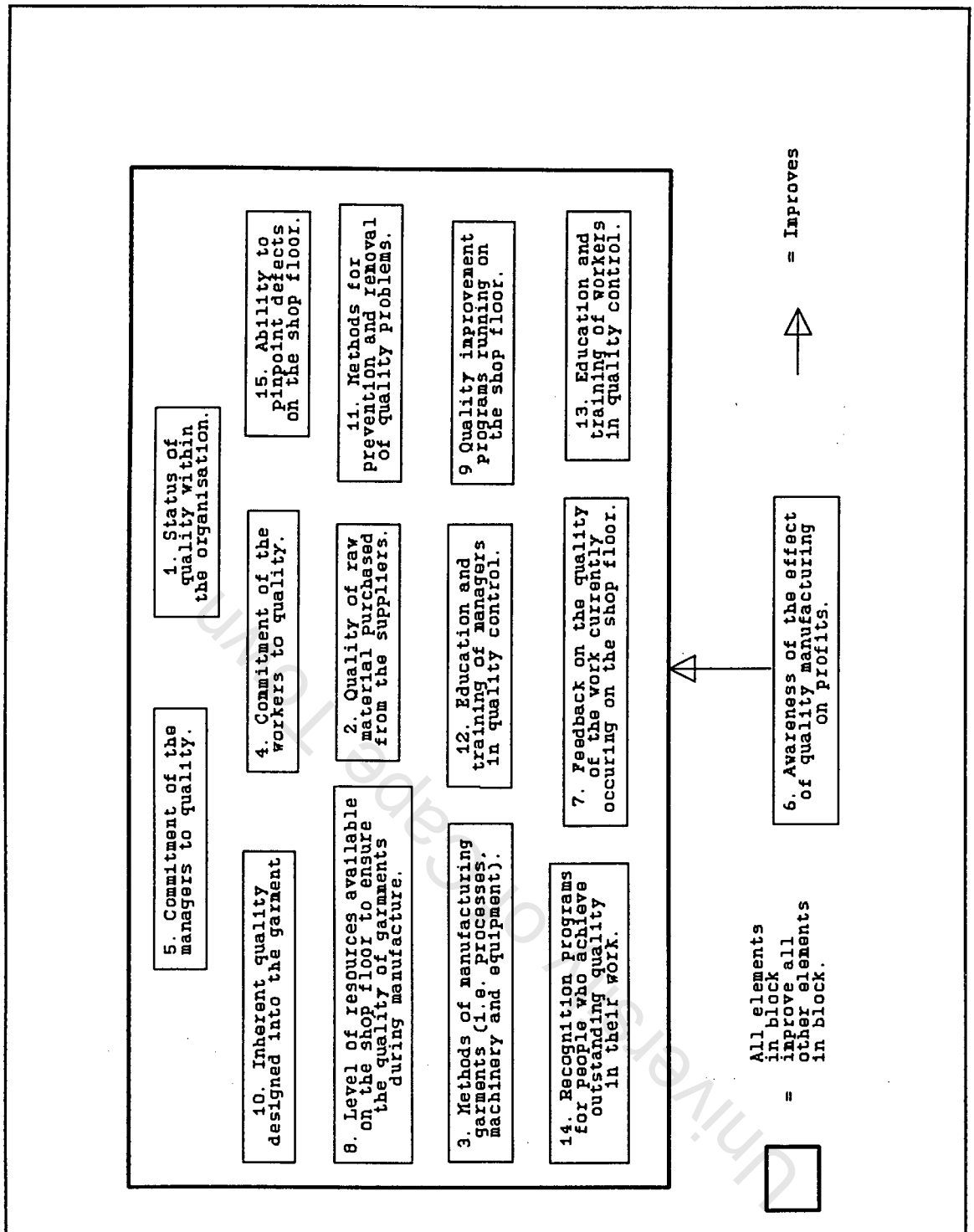
Model 5



Question and answers

1. Yes
2. No
3. No
4. Yes
5. 7
6. 12
7. 3

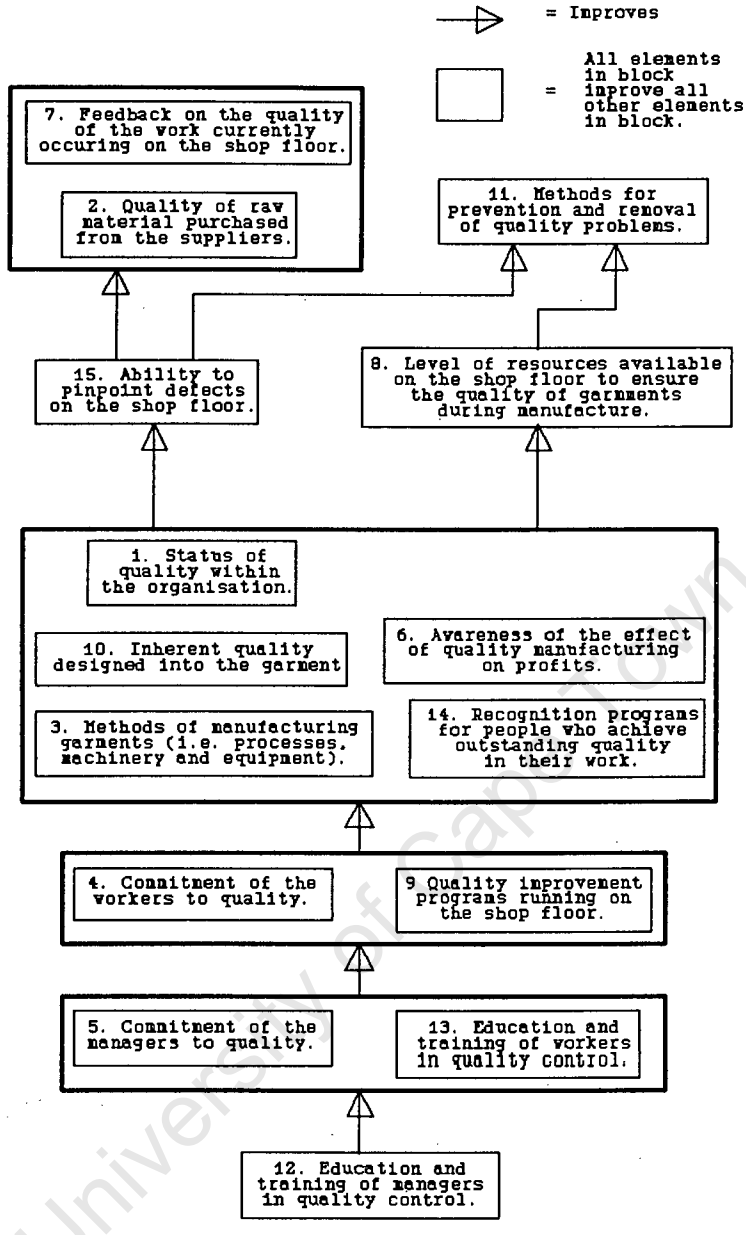
Model 6



Question and answers

1. No
2. Yes
3. No
4. Yes
5. 8
6. 20
7. 3

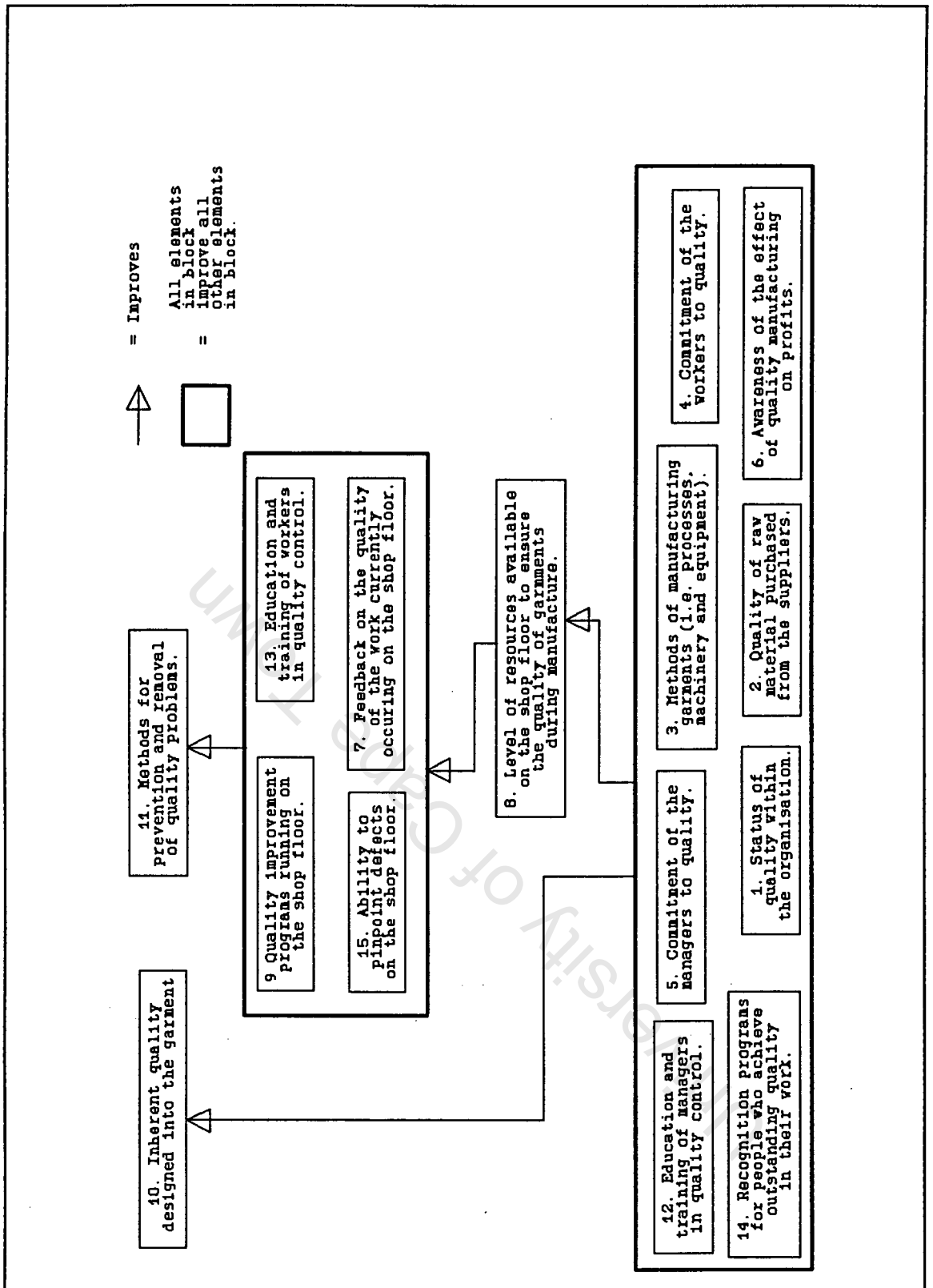
Model 7



Question and answers

1. No
2. No
3. Yes
4. Yes
5. 7
6. 4
7. 10

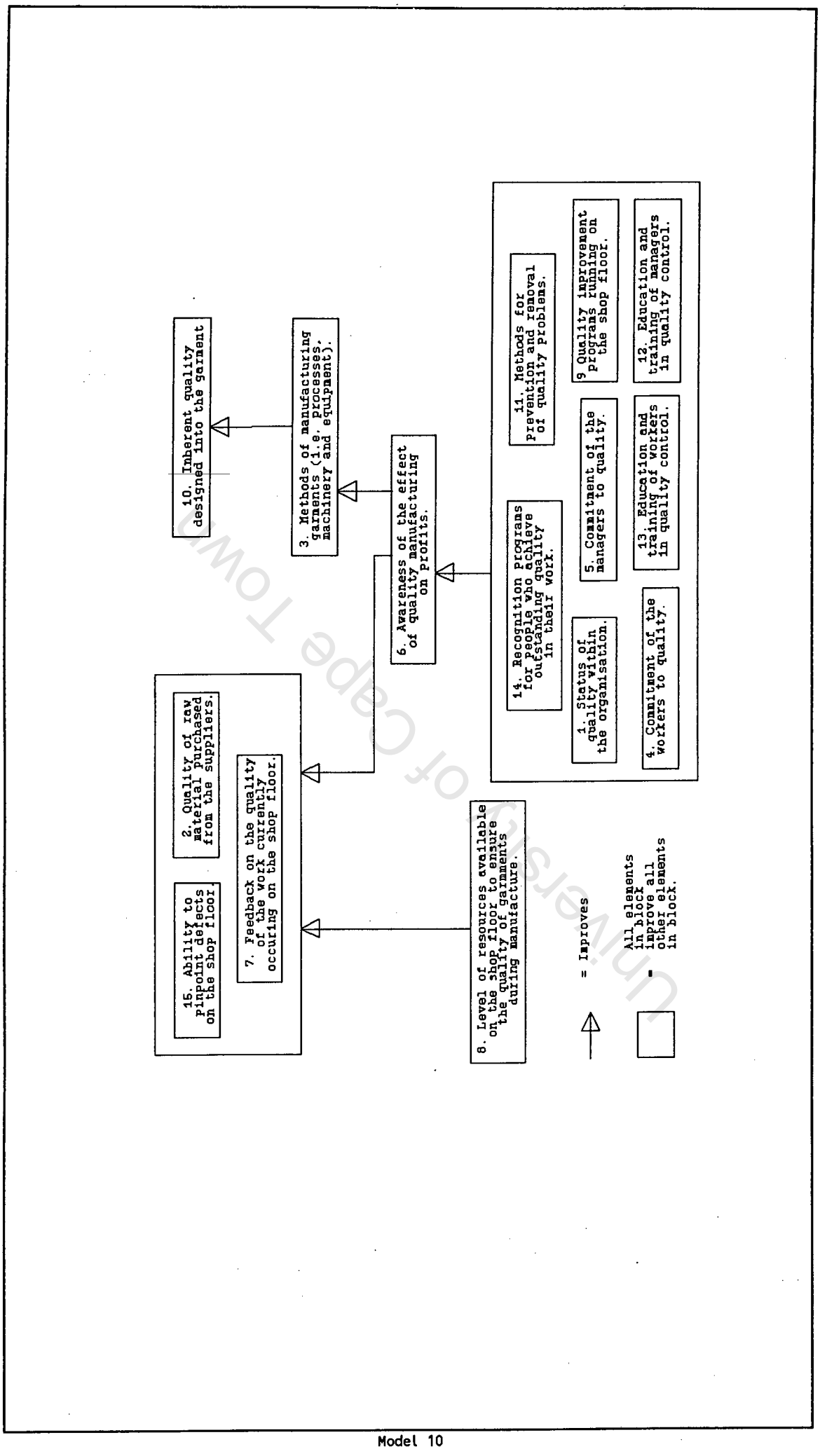
Model 8



Question and answers

1. No
2. Yes
3. No
4. Yes
5. 7
6. 12
7. 10

Model 9



Model 10

7.6. DISCUSSION OF RESULTS

The models developed are shown in figures 7.1. through 7.10. Figures 7.1. to 7.9. are the results obtained from the individual models, while figure 7.10. shows the results obtained from the group model.

It is interesting to note that many of the models have large feedback sets containing eleven or more elements. This contrast strongly with the group ISM based on the ideas of Deming. The reasons for these large feedback sets are not clear, but three hypotheses are put forward :

1. AMBIGUITY/MISUNDERSTANDING OF THE CONTEXTUAL RELATION

If the contextual relation was ambiguous, then it is possible that misunderstandings of what was being asked could have resulted. For instance, all the models with these large feedback sets show that by improving the quality of raw material purchased (element 2), the methods of manufacturing garments (i.e. processes, machinery and equipment) (element 3) is also improved. This link seems very remote, if it exists at all, and may have arisen from a badly worded contextual relation, that resulted in a misunderstanding of the question. It is interesting to note model 3, where someone who has read many books on quality management returned a model that is one large feedback set. This must surely be the result of a misunderstanding.

2. UNCERTAINTY ABOUT RELATIONSHIPS

Perhaps, if the modeller was uncertain about whether there was or wasn't a relationship between the elements, she/he gave the answer 'yes' and proceeded to the next question, without stopping longer to think the relationship through.

3. ELEMENTS NOT CLEARLY DEFINED

It is possible that the elements were not sufficiently clearly defined and that the modellers read more into each element than was intended. For instance, while doing the group ISM, the following question was raised : 'How is resources defined in the element "level of resources available on the shop floor to ensure the quality of garments during manufacture?"?' Do resources include machinery (as in element 3) and training (as in elements 12 and 13)?"

Another question that was raised in the group ISM was whether the phrase 'quality control' in elements 12 and 13 should not in fact be 'quality management'. Quality control is a process whereby defective items are discovered by inspection. Quality management is a philosophy whereby defects are prevented. This means that these two elements are not really covering the scope of ideas that they are meant to.

Model 6 is also very interesting in that two elements are not in any way connected with the rest of the model (not even to each other). This is particularly strange in that one element deals with recognition programs for workers that produce high quality goods, and improved recognition programs must surely result in improved worker commitment to quality (element 14), or/and improved status of quality in the organisation, yet this was not seen to be so by the modeller.

Model 7 also shows an interesting idea in that the awareness of the effect of quality manufacturing on profits is the base element. It is possible that the person modelling has the traditional view on quality in mind - i.e. too much quality is expensive, and there comes a point where it is not worth the extra cost. Quality is then seen as 'goodness' or 'appearance' and not as a product that meets a specification.

Models 8 and 9 are those that seem to come closest to the group model. Although model 8 has only on base element, unlike the group model, the first 3 levels of model 8 are all found on the base level of the group model. Also, elements 7,2 and 15 are inter-related in both models, though model 8 does not have them all in a feedback set like the group does.

Model 9 has five base elements common with the group model, but only one end element in common (element 10). However models 8 and 9 both have element 11 as an end element, while the group model based on Deming has it as a base element. This contrasts strongly with the group, and does not appear to be logical. It seems more logical to state that element 11 improves element 15, as is shown in the group model, as opposed to element 15 improving element 11 as shown in the other models.

7.7 CONCLUSIONS

Very little can be drawn from these models. However, the following statements seem to be valid :

1. From the group model based on Deming's ideas, it appears that the South African clothing industry does not subscribe to the ideas of Deming in its quality initiatives.

2. From the widely different models that arose from the interviews, it seems that there is no agreement of how aspects of quality management relate within the clothing industry.

3. ISM as a tool is not suited to doing interviews with individuals in an effort to obtain a models of situations.

PART IV

REFLECTIONS

8. REFLECTIONS

8.1. REFLECTIONS ON ISM

Having completed the work for the thesis, there are various thoughts and ideas that are worth noting for future reference by other researchers using the ISM methodology.

8.1.1. PHRASING OF THE CONTEXTUAL RELATION

Careful thought is needed in the choice of the contextual relation. The contextual relation used here ('would improving factor A improve factor B') seems in retrospect to be a very long winded way of saying 'would factor A enhance factor B'. The latter is not only shorter, but is easier to understand in that it requires less mental gymnastics. The better phrasing of the contextual relation could quite possibly have resulted in more meaningful models.

8.1.2. ONE-ON-ONE INTERVIEWS

It appears that ISM as a methodology is not well suited for use in an interview situation, where a single individual produces a model from a set of pre-determined elements. The reasons for this appear to be two-fold, and though at a glance the reasons may appear weak, they do in fact have far-reaching effects.

8.1.2.1. DIFFERING UNDERSTANDINGS/DEFINITIONS

Every person has their own understanding of what words and phrases mean. This is due to people's different background and learning. This difference in perception makes the comparison of models collected by the method used here of questionable value.

Each individual is reading the elements, and from their personal understanding of the words used in the elements is

answering the question. There is therefore no guarantee that different people have the same understanding of the elements. There are many different ways that one might define words like 'resources' and 'quality'. These differing definitions can result in two different people having very different ideas about what is meant by an element. Thus, though two people may answer the same question in the modelling process, in their minds they may have answered two very different questions. So each model may only be of value and meaning to the person who made it.

Also, because an individual is answering the questions for the model in silence, it is possible that the person is not, in fact, answering the question as asked by the process, but a question using the same elements, but a different contextual relation. In other words there is no assurance that the person is focusing on the relationship at hand.

Both these above-mentioned problem would be eliminated in a group situation. They would certainly be exposed by a good facilitator, who could diplomatically reveal any deviation from the contextual relation or previous meaning of the element should the group not recognise it itself. This is not true for a individual modeller, whose thoughts remain unknown. So, while the question 'will improving item A improve item B' was asked here, some modellers may have in fact been answering questions like : 'is item A important in producing high quality goods?', or 'Does item A have any effect on item B?'

This possible differing in understanding between modellers could mean that the model obtained are not comparable, and are of no value for research purposes.

8.1.2.2. THE LOSS OF STIMULATION BY DEBATE

ISM is sighted in the literature as a group process, and Hammer and Janes (ref. 18, page 11) state that ISM will help understand complexity by *increasing and clarifying the group's knowledge of the problem and its immediate environment*. However, with only one person modelling, this aspect of increased learning is lost because there is no debate. Debate challenges thought processes and helps both to remove faulty reasoning and to bring in other factors that an individual may have forgotten or be unaware of. There is no thought in creating these models external to the individual producing these models. (Some may wish to argue that the thoughts of the person who drew up the element set do influence the model.) This means there is a possibility that some very serious factors may have been ignored in modelling, and the models are very far removed from reality. The modeller has also lost out on a potentially valuable experience of learning from another person.

Another loss in learning takes place in the drawing up of the element set. It is very difficult for an individual to ensure that all elements relevant to the problem are included in the model. What if too many elements are generated to model all of them, and only a limited number can be used? The list of elements will then be those that, in the opinion of the person generating the element set, are the most relevant/important. This is very likely to effect the value of the models when drawing conclusions on how best to implement TQM in the clothing industry.

The element set is also prejudiced in that the wording of the elements is done without debate. No matter how careful one is, it is near impossible to word every element to have the same meaning to every person. This problem surfaced in the group ISM, where the question of 'What is meant by the

term "resources"?' was asked. This problem would been resolved in a group situation as the element was worded, and not only once modelling had begun.

These are two factors produce a powerful argument against the use of ISM in a 'single-modeller' situation. All the ISM literature emphasizes the group nature of ISM. However, no statements are made to deter some-one from using ISM in a single modeller situation. Thus at the start of the application, the limited value of ISM for a single modeller situation was not realised until modelling was well under way.

8.1.3. COMPARISON OF MODELS

As was mentioned earlier, the human mind can only cope with between five and nine pieces of information at any point in time. This makes the comparison of different models a very difficult task. Here each model had fifteen elements, which means that each model contains 225 pieces of information. While in this case the models were substantially different, and it was fairly easy to see that there was little in common between the models, in most instances the volume of information contained in the model is likely to present a serious problem when trying to compare them.

It is probably easiest to deal with this problem by writing a computer package that will allow some sort of comparison between the models by showing the common links, and highlighting the differences. This can be done by adding the matrices of the models together, and in inspecting the matrices every '2' entry means that there is an agreed link, ever '0' entry shows agreement in the fact that there is not a link, while a '1' entry shows a disagreement about whether a link exists between the elements. However, a matrix does not display information in an easy to read form like a digraph, and a special package would have to be written to plot the matrix with solid lines for common links, and dashed lines for disputed links. This should not prove too difficult.

In an attempt to help analyze the models, an attempt was made at adding together the matrices from the various models in order to try to create a 'group' ISM from the individual models. However, this is not an easy task for the reason outlined below :

Assuming that the following three sub-matrices make up part of the models created by three individuals :

	C	B	A
C	1	0	0
B	1	1	0
A	1	1	1

	C	B	A
C	1	0	0
B	1	1	0
A	0	0	1

	C	B	A
C	1	0	0
B	0	1	0
A	0	1	1

(8.1)

These three matrices would be added together to give the following matrix :

	C	B	A
C	3	0	0
B	2	3	0
A	1	2	3

(8.2.)

Assuming that a majority vote is sough in order to create the matrix for the group ISM. Every '2' and '3' in eq. 8.2. can be replaced by a '1', and every '1' and '0' can be replaced by a '0'. This will result in the matrix below :

	C	B	A
C	1	0	0
B	1	1	0
A	0	1	1

(8.3.)

However, this is not a reachable matrix as although A relates to B, and B relates to C, A does not relate to C. This is mathematically unsound, and the model is meaningless. There is no sound method of making the model reachable.

One possibility explored in trying to create a group ISM from a series of individual models was the following :

It can be argued that the matrix shown in eq. 8.3. is not valueless. If a group consisting of the same individuals who made the individual models had been in the room together, and they had voted on the questions as they were asked, a model would have been created. Thus a 'reconstruction' of what the group would have answered can be made by answering the questions asked on the basis of the answers in matrix 8.3., while running the ISM package with the same element set.

Although this 'reconstructed matrix' will differ from that of 8.3., it will be that matrix that would have been made had the group been in the room together. Also, although matrix 8.3. is not reachable, the modelling processes will imply answers and the 'reconstructed matrix' will be a reachable matrix.

This logic is faulty for two reasons :

1.It assumes that no matter what might have been said between the individual modellers had they been in a room together, no-one would have been persuaded to vote differently to how they ha voted initially.

2. If the questions posed were not all asked in the same order to each individual, and asked in that same order when reconstructing the matrix, then the model generated by this method is nothing more than one of many possible reachable models that can be generated from the grouping of the matrix. Certain ISM packages (such as the one written here) will ask the initial questions in the same order, before the mathematics takes over to find the best questions to ask. Thus the 'reconstructed matrix' may have some value. However, should the initial question's be asked in a random fashion, the 'reconstructed matrix' will be quite valueless.

No method is known for overcoming this problem, and it appears that comparing different models made from the same element set will always be a problem.

8.2. REFLECTIONS ON TQM

TQM is undoubtedly on of the keys to the success of Japanese industry. The main concepts of TQM as listed in section 6.3. are all important and fundamental to a TQM initiative. However, from my limited experience in industry, there are several key elements of ISM that stand out.

1. The commitment of top managers to the TQM initiative. Not only must the commitment be there in words, but it must be one that is there in action. Speaking about quality, and yet purchasing low quality (or reject) raw material does not place a quality initiative on a firm footing. Thus top-level managers must see their role within the quality initiative and be open to hear the opinion of their sub-ordinates on what needs to be changed, and then change it.

2. The implementation of TQM is going to cost money. There is an entire work-force that has to be trained to understand what is expected of them. It does not help to go to a factory worker and tell him or her to produce quality goods. Quality has to be defined for a worker in the terms he or she can understand. Once this is clear, the workers must be trained to operate the machinery properly, and on what methods are being used to record data, report faults, make suggestions, etc. etc.

   This means spending money and losing production for a period of time while training the work-force. A company that is not prepared to lose a little in the short-term for a major gain in the long-term will never transform itself to produce quality.

3. All employees are under the same umbrella when it comes to quality of work. I know of an instance where a company totally ran out of one type of raw material necessary for production of 40% of its range of goods. This mistake probably cost of the order of several million Rand in lost turnover. Now, normally this company would fire a factory worker who produced a day's worth of reject. However, all that happened to the person placing (or not placing) the orders was a verbal reprimand - yet his mistake cost the company far more then several tons of reject. It is easily apparent that all the employees are not under the same umbrella. (This is probably on of the reasons for the rise of trade unions in South Africa.)

   This same incidence has most certainly occurred in the field of quality management, where production lines must make quality, but the failure to order spare parts is normal. TQM means making the entire organisation into a team, and breaking down the barriers that separate the 'workforce' and the 'managers'.

4. It is no use having the quality department reporting to the head of production. This is a disastrous way of running trying to implement TQM. It is tantamount to saying that the production head is producing goods of acceptable quality to the top managers. However, if the waste rates are really too high, the person who must know is the production manager's boss (and, of course, the production manager) in order to put pressure on the production manager to improve matters. If the information on the quality of product never goes beyond the production manager, then he is likely to aim at quantity to the expense of quality, because the goods must be out on time.

   It is frustrating to see production managers tell people to run machines when the quality of the product is below par. Problems are not solved by being made aware of them. Problems

are solved by the expenditure of effort to solve them. It seems that production managers are often not aware of this, and are happy to say something like 'Yes, I know the waste on the run was high. I was told we had a problem when the run started, and I told them to go ahead as we had to meet the delivery schedule.' What is even more amazing is that statements like that tend to get a response like the following: 'Oh, well as long as you were aware of it that's OK.' instead of 'If it was wrong why didn't you tell the customer it was going to be late and fix the problem. I don't want waste rates like this ever again!'

From talking to the various people I interviewed, it seems that the South African clothing industry, in general, is not particularly aware of the TQM philosophy. This is clearly reflected in the answers to the background questions.

It seems that quality is perhaps a bit difficult to define for the clothing industry. Unless there is very expensive colour-matching equipment available, it is very difficult to determine whether the colour of the trim is close enough to what was requested. It becomes a matter of subjective judgement.

It is also difficult to measure a lot of the quality aspects in clothing. To measure stitch length is not easy, as threads are generally colour-matched to the fabric, and thus difficult to see. How easy and quick is it to measure collar-size, waste size, inside leg? Is it that important that everything is correct, as people are so variable? I think that these thoughts pervade the industry into seeing quality as being that which the consumer can see as a fault. The customer will not complain if there are 10 or 12 stitches per centimetre so it is not important. However, stitches should not unpick, as this will cause complaint.

The main driving force in the clothing industry seems to be one of processability. As long as the good process easily, and do not hook, catch, or pull out, most companies are quite content, as

throughput and thus turnover is high. One of the major traits of the clothing industry seems to be that everything is always late. This is possibly one reason why processability is so important, because the delivery date has come and gone, and the customer is complaining.

Even though certain of the big chain-stores have there quality standards, and can be very fussy about the goods that are received. General perception within the industry is that when some-one is desperate they will take anything! This is not a healthy environment for thoughts and ideas on quality management to grow.

8.3. REFLECTIONS ON COMPLEXITY

Complexity is part of life. By far the majority of problems faced daily are difficulties, rather than messes. However, at any point in time there is probably a messy problem facing any person in the world. It may be fair to say that a large percentage of people never see their problems as being messes, but only see the problem as a difficulty. This is a gross generalisation, but most people have a 'now' kind of mentality, and fail to see the long-term effect of actions. (This is especially true of people who keep buying goods on hire purchase, instead of realising the saving of paying cash and not incurring interest on their debt.) If some-one is unable to see more than the short-term on this kind of issue, can one expect them to see the greater whole of a messy problem?

Also, because man has a limit of storing 7±2 chunks of information, most people have no hope of dealing with messy problems without being able to break them up into parts. Thus many problems are probably only ever seen as several unlinked difficulties, when in reality they are messes. In support of this one can consider the of thousands of companies world-wide who have a quality 'problem', and yet have no strategy to solve it. They see the 'problem' as a minor difficulty, that doesn't really

need to be solved - yet. However, reality is that it is a very messy problem that is costing a fortune, harming the company image, and producing dissatisfaction in the work-force. If managers of these companies could see the real situation, they would have called in the experts years ago.

One could argue that the companies such as those mentioned above have been overcome by the mess to the point at which they do not know where to begin. This argument is flawed in that it is generally possible to call an outside 'expert' to help solve problems, and this has not been done.

8.4. REFLECTIONS ON RESEARCH AND ENQUIRY

Research and enquiry is a tool for helping to deal with complexity, whether it be in arts, marketing, engineering or finance. Knowledge and understanding are the great keys to advancement.

I think that for any future research I undertake, more time will be spent on planning the route before embarking upon it. Definite and well defined goals and mile-stones will be in place, and aimed at with purpose. Combined with that is a definite need to keep good records and a steady pace at writing up sections as one goes along.

On a more broad basis, it is a very humbling experience to learn so much. It makes one realise how great a mind some-one like Newton, Galileo, Freud and other well-known researcher must have had to leave behind the work that they have. The more that is learnt, the more one sees how much there is still to learn. This thesis adds but a drop to the sea of knowledge.

9. REFERENCES

1. Dale, B. and Cooper, C. (1992) *Total Quality and Human Resources*, Blackwell Publishers, Oxford, England.

2. Deming, W.E. (1991) *Out of the Crisis*, Cambridge University Press, Cambridge, England.

3. Crosby, P.B. (1984) *Quality Without Tears*, McGraw-Hill, New York, USA.

4. Crosby, P.B. (1979) *Quality is Free*, McGraw-Hill, New York, USA.

5. Juran, J.M. (1989) *Juran on Leadership for Quality*, Free Press, New York, USA.

6. Juran, J.M. (1992) *Juran on Quality by Design*, Free Press, New York, USA.

7. Ishikawa, K. (1985) *What is Total Quality Control? The Japanese Way*, Prentice-Hall Inc., London, England (Translated by David J. Lu.).

8. Flood, R.L. (1993) *Beyond TQM*, John Wiley & Sons, Chichester England.

9. Price, F. (1990) *Right Every Time*, The Camelot Press Ltd, Southampton, England.

10. Barret, D. (1994) *Fast Focus on TQM*, Productivity Press Inc., Portland, USA.

11. George, S. and Weimerskirch A. (1994) *Total Quality Management*, John Wiley & Sons, New York, USA.

12. Warfield, J.N. (1989) *Societical Systems - Planning, Policy and Complexity*, Intersystems Publications, USA.

13. Moore, C.M. (1987) *Group Techniques for Idea Building*, Sage Publications Inc.

14. Bounds, G., Yorks, L., Adams, M., Ranney, G. (1994) *Beyond Total Quality Management, Toward the emerging paradigm*, McGraw-Hill Book Co. Singapore.

15. Gower, (1993) *Handbook of Management*, 3rd Edition, Gower Publishing Ltd, U.S.A.

16. Open University (1990) *Managing in Organisations*, part of *Technology : A Second Level Course*, Open University Press, England.

17. Jeffery, R.J. and Janes, F.R. *The use of Interpretive Structural Modelling in planning for a community speech therapy service*, taken from *Proceedings of the 33rd Annual Meeting of the International Society for the Systems Sciences*, Edinburgh, July 1989, Vol. 2, p314-319.

18. Hammer, K. and Janes, R. *Interactive Management - Structural Mapping integrates ideas and improves group decision making*. From *Operational Research Insight*, Vol. 3, No. 1, January-March 1990.

19. Janes, F.R. *Planning for an industrial training board team : an application for interactive management*. From *Proceedings of the International Symposium on Communication, Meaning and Knowledge vs. Information Technology*, Lisbon, 13-15 September 1989, Vol. 1, Paper 37, pp1-6

20. Janes, F.R. *Interpretive Structural Modelling : a methodology for structuring complex issues*. Trans. Inst M C, Vol. 10, No. 3, August 1988.

21. Saunders, C.S. *Structural modelling in a corporate environment*, Tandem Communications.

22. Bell, J., Bush, T., Fox, A., Goodey, J., Goulding, S. (1984) *Conducting Small-scale Investigations in Educational Management*. Harper & Row Ltd. London, England.

23. Dillon, J.T. (1990) *The Practice of Questioning*. Routledge, London, England.

24. Calvert, C. (1993) *Turbo Pascal Programming 101*. SAMS Publishing, Indiana, U.S.A.

25. *Turbo Pascal 7.0 - Programmer's Reference*. Borland International, California, U.S.A.

26. *Turbo Pascal 7.0 - Language Guide*. Borland International, California, U.S.A.

27. *Turbo Pascal 7.0 - User's Guide*. Borland International, California, U.S.A.

28. Warfield, J.N. (1990) *A Science of Generic Design. Managing complexity through systems design. Vol I & II*. Intersystems Publications, U.S.A.

29. Simon, H.A. (1974) *How big is a chunk?* 'Science', 183(8), 482-488, referenced in Janes[20]

30. Miller, G.A. (1956) *The magical number seven, plus or minus two: some limits to our capacity for processing information'*, 'Psych Rev, 63(2), 81-97. referenced in Janes[20]

31. World Economic Forum, (1994) *The World Competitiveness Report 1994*.

32. Checkland, P. (1981) *Systems thinking, systems practice*, The Bath Press Avon, England.

33. Flood, R.L., and Carson E.R. (1993) *Dealing with complexity - an introduction to the theory and application of systems science*. Plenum Press, New York, U.S.A.

34. African National Congress (1994) *The Reconstruction and Development Programme - A policy framework*. Umanyano Publications, Johannesburg, South Africa.

35. Logothetis, N., (1992) *Managing for Total Qualityy - from Deming to Taguchi and SPC*. Prentice Hall International (UK) Ltd.

APPENDIX A

THE MATHEMATICS OF ISM

ISM relies on finite or discrete mathematics to choose the best questions to ask. (i.e. Those questions whose answers will result in the most implied answers.) ISM in fact uses a branch of mathematics referred to as Boolean algebra. (i.e. Only the numbers 0 and 1 exist in the number line.) Boolean mathematics lends itself to use in fields such as logic and computers, and so is aptly suited for ISM.

ISM uses set theory and matrices extensively. Matrices exceeding 100 rows by 100 columns have to be raised to the power of two or higher, often in iterative processes. This is the primary reason for ISM modelling needing the use of a computer.

The mathematics shown here is taken from Warfield[12], and readers seeking more insight into the theory are directed there. However this section covers all aspects from the start of modelling to the plotting of the digraphs.

There are two foundations on which ISM is laid in order to get to the matrix for the model. They are the *coupling method* and the *scanning method*. Generally, the scanning method is used first in modelling, following which the coupling method finishes off the modelling process. Both these processes shall be explained in this section, along with the method of obtaining the digraphs, once the matrix has been completed.

A.1.1. THE SCANNING METHOD

The scanning method is used in a situation where no prior modelling has been done, or new elements are being added to a complete model. The first phase in the method results in the matrix being partitioned on an element (the so-called 'partitioning on elements'), following which the coupling method is used to 'fill in' the rest of the unknowns.

The partitioning on elements is a cyclical process, where the number of parts in each cycle fluctuates, but generally

initially increases, and then decreases. Each part is processed by the same four steps.

In this section, let :

M be the matrix to be partitioned.
U represent the total set of element to be modelled.
u represent any arbitrarily chosen element.'
nxn be the size of *M*.
R be the contextual relation (e.g. 'precedes')

STEP 1.

The element *u* for partitioning is chosen.

STEP 2.

Split the elements of the set *U-u* into two subsets. The first is called the 'yes set' ($\dot{Y}(u)$), and second is called the 'no set'. The yes set consists of all elements of *U-u* that are succedent to *u*, and the no set consists of the rest of the elements that are not succedent to *u*.

This splitting of the elements is done quite easily. The question 'element *u* R element *x*' $\{x \in (n-u)\}$ is posed. Where the answer 'yes' is received, the element is placed in the yes set, and a '1' is inserted into the matrix at the relevant point. Similarly, the answer 'no' places the element in the no set and a '0' is inserted in the matrix.

Thus the set *U* can be partitioned as follows :

$$\prod(U) = Y(u); \quad u; \quad U\text{-}Y(u)\text{-}u$$

STEP 3.

The yes set is now partitioned into two subsets. The first is the 'lift set' ($L(u)$), and consists of those elements of $Y(u)$ that are NOT antecedent to u. The other subset, is called the 'feedback set' ($F(u)$) consists of those members of $Y(u)$ that are antecedent to u. The feedback set are all in a cycle set with u. That is to say that u R $F(u)$ and $F(u)$ R u.

This partitioning is also easily done, by posing the question 'does element x R element u' $\{x \in Y\}$. Where the answer 'yes' is received, the element is antecedent to u, and is placed in the feedback set, and a 1 is placed in the matrix. The answer 'no' indicates that the element is not antecedent to u, and so the element is placed in the lift set, and a 0 is placed in the matrix.

STEP 4.

This step consists of partitioning the no set into two subsets. The one is the 'vacancy set,' ($V(u)$), and consists of those elements that are NOT antecedent to u. None of the members of the vacancy set connect to u. The other set is the 'drop set,' ($D(u)$), and contains the elements that are antecedent to u.

The same process as in step 3 is used to identify the two sets, with the elements receiving a 'yes' answer going into the drop set, and the others into the vacancy set.

Thus the following partition of the set U can be found :

$$\prod(U) = L(u); \quad F(u); \quad u; \quad V(u); \quad D(u)$$

From this information, the matrix M can be sorted and displayed in the form shown in figure A.1.

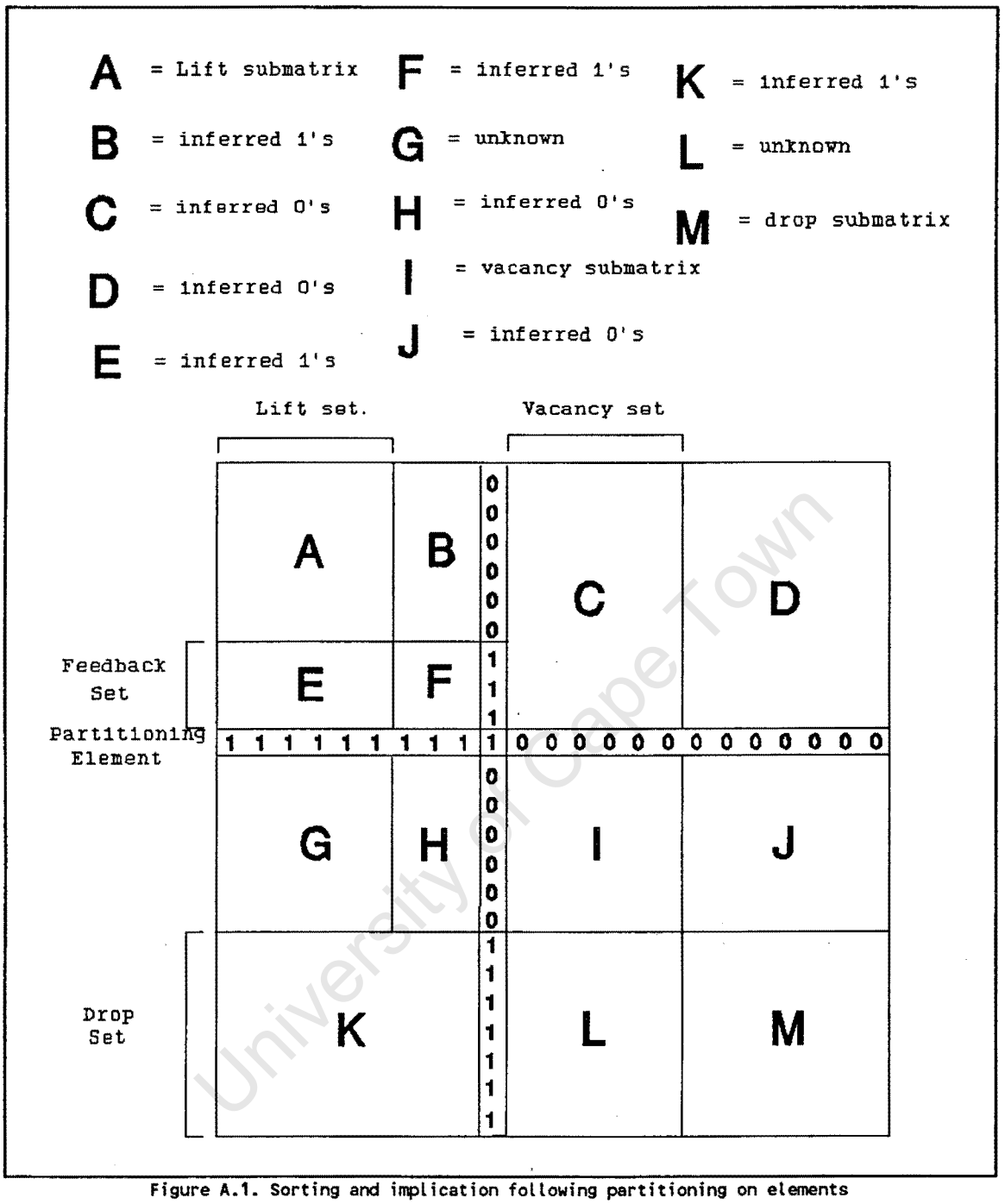


Figure A.1. Sorting and implication following partitioning on elements

From the information generated from these few questions and answers, a large number of answers can be implied. In general there are four submatrices filled with inferred 0's, and four submatrices filled with inferred 1's. With reference to figure A.1. they are the following :

Considering submatrix C : *u* is antecedent to the lift and feedback set. So, if any member of the lift or feedback sets is also antecedent to the vacancy set, *u* also must be antecedent to the vacancy set. This is a contradiction of the definition of the vacancy set, so C is filled with inferred 0's.

Considering submatrix D : No element of the lift or feedback sets can be antecedent to the drop set. Each element of the drop set is antecedent to *u*, and so if a member of the lift or feedback set was antecedent to any member of the drop set, the element in the drop set would be in the feedback set, and not in the drop set.

Considering submatrices B,E and F : The feedback set is in a cycle set with element *u*. That is to say that both are antecedent to each other. They must therefore have identical interconnections with all elements in the matrix. Therefore E and F are filled with inferred 1's, and B with inferred 0's.

Considering submatrix H : As was stated above, the feedback set has the same interconnections as the element *u*. Therefore, as the vacancy set does not connect to *u*, it does not connect to the feedback set either. So, H is filled with inferred 0's.

Considering submatrix J : All members of the drop set are antecedent to *u*. If any member of the vacancy set was antecedent to any member of the drop set, it would therefore also be antecedent to *u*. This is contradiction of the definition of the vacancy set, and so J is filled with inferred 0's.

Considering submatrix K : All members of the drop set are antecedent to *u*. However *u* is antecedent to all members of the lift and feedback sets. Therefore all element of the drop set are also antecedent to all members of the lift and feedback sets, and K is therefore filled with inferred 1's.

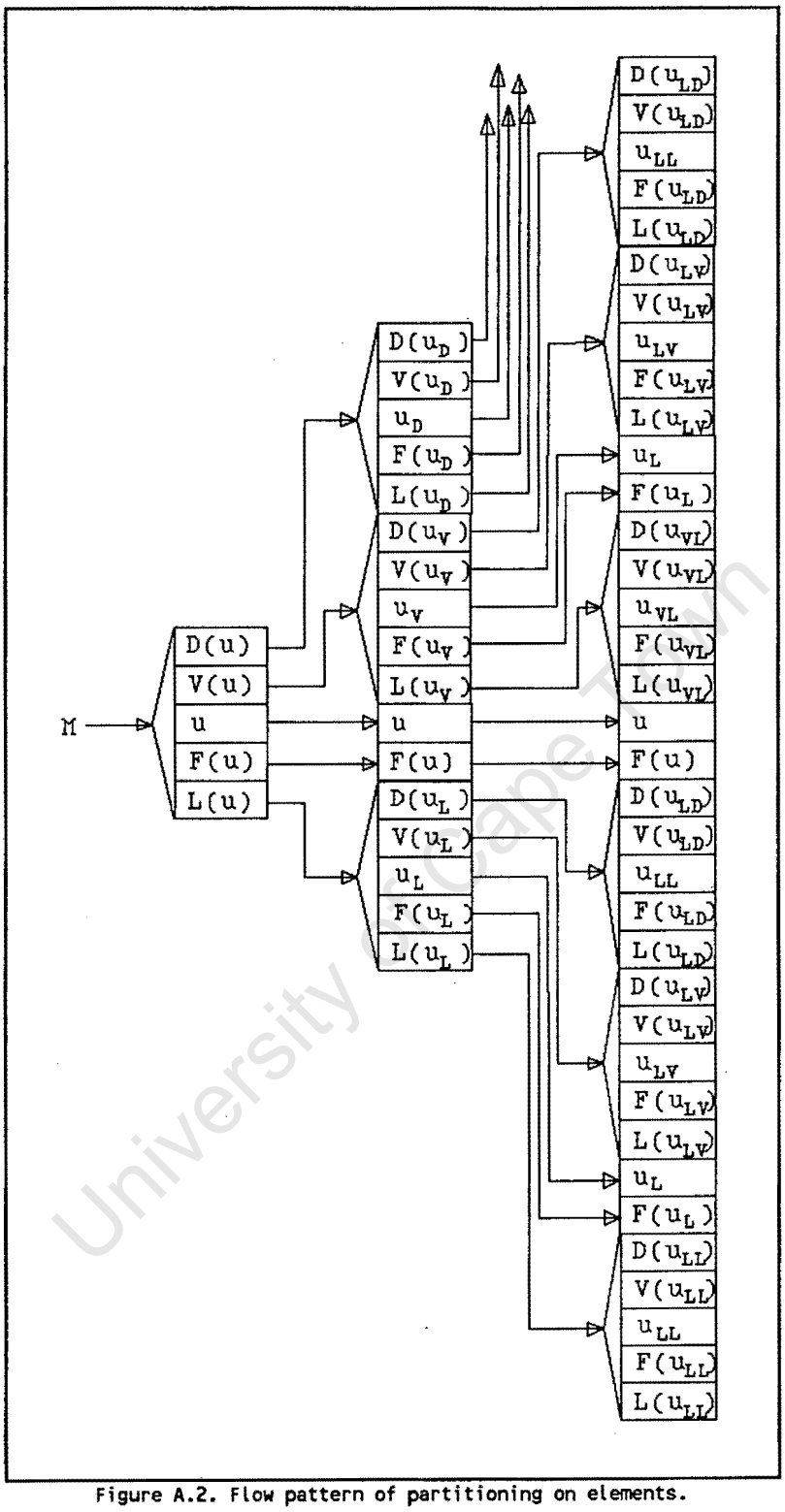


Figure A.2. Flow pattern of partitioning on elements.

The second cycle of partitioning on elements is now begun. Matrix M in the first cycle is replaced with the lift set - i.e. the submatrix A shown in figure A.1. U is replaced by $L(u)$, and a new u_L is arbitrarily chosen as before. Once the steps one to four have been completed for the lift set, and the inferred answers have been filled in, the vacancy set and the drop set are each taken in turn and inserted for matrix M in the process, just as was the lift set. (Should any submatrix be of the size of one or smaller, it is not processed through any of the four steps.)

The third cycle is then begun in a similar way, with the lift set of the original lift set being the first to be modelled, followed by the vacancy set of the original lift set, and continuing until the drop set of the original drop set is modelled. This results in a pattern as shown in figure A.2. The process continues through as many cycles as it takes until all submatrices are of size one or zero. Once this has occurred, the partitioning by elements is complete, and the second stage of coupling matrices can begin.

A.2. THE COUPLING METHOD

After a matrix has been partitioned by elements, the unknown submatrices (shown as G, and L in figure A.1.) of each cycle still have to be found. However, in filling in these submatrices it is desirable to still imply as many answers as possible. This is why interconnection theory is used to ensure that the most answers possible are implied. This section first deals with the mathematics of interconnection theory, before giving a worked example for illustration.

A.2.1 INTERCONNECTION THEORY

If any two adjacent submatrices are considered, they can be written in the general form :

$$M = \begin{bmatrix} D & S \\ T & E \end{bmatrix}$$

where M is a square matrix, D and E are known square submatrices, not necessarily of the same dimensions, and S and T are unknown submatrices, not necessarily square.

Since M must be a reachability matrix for ISM to work, $M^2 = M$. Hence the following constraints must be fulfilled :

$$\begin{aligned} D^2 + ST &= D \\ DS + ST &= S \\ TD + ET &= T \\ ST + E^2 &= E \end{aligned} \qquad \text{(EQ. A.1.)}$$

Since D is reflexive, $D^2 \geq D$, thus from equation B1 $D^2 = D$. Hence D must be a reachability matrix. Similarly EB is a reachability matrix. Also, as D is reflexive, $DS \geq S$. Thus from equation B1, $DS = S$, by similar reasoning, $SE = S$, $TD = T$, and $ET = T$. Thus equations B1 can be replaced by the following constraints :

$$
\begin{aligned}
ST &\leq D \\
DS &= S \\
SE &= S \\
TD &= T \\
ET &= T \\
TS &\leq E
\end{aligned}
\qquad \text{(EQ. A.2.)}
$$

From further manipulation of these constraints, a complete implication matrix for the unknowns S and T can be found. This is summarised in the following three step procedure, which is programmable into a computer :

STEP 1.

Form the matrix :

$$
N_1 = \begin{bmatrix} E_T & \phi \\ \bar{E}+I & E_T \end{bmatrix} \qquad \text{(EQ. A.3.)}
$$

where ϕ is the zero matrix.

STEP 2.

Make the following substitutions in N_1 to arrive at a matrix N_2 :

D for each 1 on the main diagonal in E^T.
$\bar{D}_T$ for each main diagonal 1 in $\bar{E}+I$.
I for all other entries of 1 in N_1.
ϕ for all 0's in N_1.

STEP 3.

Find the reachability matrix Φ of the matrix N_2 by raising it to powers. (I.e. Keep raising N_2 to a higher and higher power until $N^{n+1} = N^n$. $\Phi = N^n$.) The matrix Φ is a complete implication matrix for S and T, and is indexed by the set

Z, where Z is the complement of the unknowns of S plus the unknowns of T.

It can be seen that Φ is the magnitude of 2*(size of D)*(size of E). So if D and E are both of size ten, N_2 is a matrix 200 by 200, that has to be progressively raised to powers until $N^{n+1} = N^n$. (ϕ is the same size as N_2.)

Once Φ has been found, it is possible to determine what unknowns in S and T can be inferred should certain entries become known. This is because if any entry in Φ takes on the value 1, then every element in its succedent set must also take on the value 1. Also, if any entry takes on the value 0, every element in its antecedent set does too. The succedent and antecedent sets are easily found by the 1 entries in the Φ matrix.

The *inference opportunity* of any element in Z can be defined as an ordered pair (α,σ), where α is the number of unknowns that will take the value 0 should the element take the value 0, and σ is the number of unknowns that will take the value 1 should the element take the value 1. So, α is effectively obtained by counting the 1's in the column of Φ corresponding to the element, while σ is obtained by counting the 1's in the row of Φ corresponding to the element. Both α and σ should be reduced in value by 1 as the element itself is included in the total obtained.

For any element, α and σ will probably differ in value from each other, and will also differ between elements. The best element to obtain an answer for is that one which will guarantee the highest number of inferred answers. That means finding the lower of α and σ for each element, and then choosing the element for which that is the highest.

Obviously, as answers are inferred, the inference opportunity decreases. As values for elements become known

(either directly by the answer to a question, or from an inferred answer to a question) the rows and columns for these elements are deleted from Φ which decreases in size. Each time the size of Φ is decreased, the inference opportunity is recalculated, and a new element is chosen for which an answer is sought. This process is illustrated in the following worked example.

A.2.2 EXAMPLE OF COUPLING METHOD

This example is designed to illustrate the coupling method.

Let the matrix M be given as :

1	0	s_1	s_3	s_5
1	1	s_2	s_4	s_6
t_1	t_2	1	0	0
t_3	t_4	0	1	0
t_5	t_6	1	1	1

Evidently :

$D =$

1	0
1	1

$S =$

s_1	s_3	s_5
s_2	s_4	s_6

$T =$

t_1	t_2
t_3	t_4
t_5	t_6

$E =$

1	0	0
0	1	0
1	1	1

Let the element set for D consist of $\{D1,D2\}$, and the element set for E consist of $\{E1,E2,E3\}$.

From equation B3 N_1 can be obtained as follows :

$N_1 =$

1 0 1 0 1 1 0 0 1	0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 0 0 1	1 0 1 0 1 1 0 0 1

(EQ. B4)

From N_1, D can be substitute in to obtain N_2 :

$$N_2 = \left[\begin{array}{cccccc|cccccc} 1&0&0&0&1&0 & 0&0&0&0&0&0 \\ 1&1&0&0&0&1 & 0&0&0&0&0&0 \\ 0&0&1&0&1&0 & 0&0&0&0&0&0 \\ 0&0&1&1&0&1 & 0&0&0&0&0&0 \\ 0&0&0&0&1&0 & 0&0&0&0&0&0 \\ 0&0&0&0&1&1 & 0&0&0&0&0&0 \\ \hline 0&0&1&0&1&0 & 1&0&0&0&1&0 \\ 1&0&0&1&0&1 & 1&1&0&0&0&1 \\ 1&0&0&0&1&0 & 0&0&1&0&1&0 \\ 0&1&1&0&0&1 & 0&0&1&1&0&1 \\ 0&0&0&0&0&0 & 0&0&0&0&1&0 \\ 0&0&0&0&1&0 & 0&0&0&0&1&1 \end{array}\right] \qquad \text{(EQ. A.5.)}$$

The complete implication matrix Φ is now found by raising N_2 to progressively higher powers until $N_2^{n+1} = N_2^{n}$:

$$\Phi = \left[\begin{array}{cccccc|cccccc} 1&0&0&0&1&0 & 0&0&0&0&0&0 \\ 1&1&0&0&1&1 & 0&0&0&0&0&0 \\ 0&0&1&0&1&0 & 0&0&0&0&0&0 \\ 0&0&1&1&1&1 & 0&0&0&0&0&0 \\ 0&0&0&0&1&0 & 0&0&0&0&0&0 \\ 0&0&0&0&1&1 & 0&0&0&0&0&0 \\ \hline 0&0&1&0&1&0 & 1&0&0&0&1&0 \\ 1&0&1&1&1&1 & 1&1&0&0&1&1 \\ 1&0&0&0&1&0 & 0&0&1&0&1&0 \\ 1&1&1&0&1&1 & 0&0&1&1&1&1 \\ 0&0&0&0&0&0 & 0&0&0&0&1&0 \\ 0&0&0&0&1&0 & 0&0&0&0&1&1 \end{array}\right] \qquad \text{(EQ. A.6.)}$$

The index set for Φ consists of the set :

$$Z = \{z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8, z_9, z_{10}, z_{11}, z_{12}\}$$

where :

$$\begin{aligned} z_1 &= \bar{s}_1 \\ z_2 &= \bar{s}_2 \\ z_3 &= \bar{s}_3 \\ z_4 &= \bar{s}_4 \\ z_5 &= \bar{s}_5 \\ z_6 &= \bar{s}_6 \\ z_7 &= t_1 \\ z_8 &= t_2 \\ z_9 &= t_3 \\ z_{10} &= t_4 \\ z_{11} &= t_5 \\ z_{12} &= t_6 \end{aligned} \qquad \text{(EQ. A.7.)}$$

The initial inference opportunity for Φ is shown in table A.1.

z_1	=	(4,1)
z_2	=	(1,3)
z_3	=	(4,1)
z_4	=	(1,3)
z_5	=	(10,0)
z_6	=	(4,1)
z_7	=	(1,3)
z_8	=	(0,8)
z_9	=	(1,3)
z_{10}	=	(0,8)
z_{11}	=	(5,0)
z_{12}	=	(2,2)

TABLE A.1. - Initial inference opportunity for ϕ.

From table A.1. it can be seen that the best element choice is z_{12}, as it guarantees a minimum inference of two

elements. With reference to equation B7, $z_{12} = t_6$, which is the answer to the question 'does E_3 enhance D_2 ?' This is therefore the best question to ask. Assuming that the answer is yes, then z_5, z_{11} and z_{12} all take on the value 1. As $z_5 = \bar{s}_5$, s_5 takes on the value 0, while t_5 and t_6 take on the value 1. The rows and columns of these elements are deleted from Φ and Φ_1 is as follows :

$$\Phi_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

From Φ, a new table of inference opportunities for the remaining elements can be constructed as per table A.2.

z_1	=	(4,0)
z_2	=	(1,2)
z_3	=	(4,0)
z_4	=	(1,2)
z_6	=	(4,0)
z_7	=	(1,1)
z_8	=	(0,5)
z_9	=	(1,1)
z_{10}	=	(0,5)

TABLE A.2. - Inference opportunity for Φ_1.

As can be seen from table A.2., the choice between z_2 and z_4 is arbitrary. However assuming that z_4 is chosen and that D_2 does not enhance E_2, x_4 *will take the value 0, and* z_4 *will take the value 1.* So, z_3, z_4 and z_6 are all 1's, and x_3, x_4 and x_6 are all 0's.

Φ_2 is now found by removing the rows and columns of z_1, z_4 and z_6, and a new table of inference is computed. So far only two questions have been asked of the modeller, while six entries have been filled in the matrix M. This equates to only 33% of the answers having been directly asked of the modellers, with the remainder being implied by the mathematics.

$$\Phi_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

z_1	= (4,0)
z_2	= (1,1)
z_7	= (1,0)
z_8	= (0,2)
z_9	= (1,1)
z_{10}	= (0,3)

TABLE A.3. - inference opportunity for Φ_2.

From table A.3. it is evident that the choice between z_2 and z_9 is arbitrary. If z_9 is chosen, and E_3 enhances D_1, then z_9 and z_1 have the value 1. So, x_1 is 0, and y_3 is 1. Repeating the process of finding a new matrix and table, from table A.4. it is evident that the choice for the next question is once again arbitrary. Assuming that z_{10} is chosen, and that E_2 enhances D_2, z_{10} and z_2 become 1's. So x_2 is a 0, and y_4 is a 1.

$$\Phi_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

z_2	= (1,0)
z_7	= (1,0)
z_8	= (0,1)
z_{10}	= (0,1)

TABLE A.4. - Inference opportunity for Φ_3.

Φ_4 is now found, and so is the new table of inference opportunities.

$$\Phi_4 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

z_7	= (1,0)
z_8	= (0,1)

TABLE A.5. - Inference opportunity for Φ_4.

Thus, assuming that the question 'does E_1 enhance D_2' has the answer 'yes', both z_7 and z_8 have the value 1, as do y_2 and y_3. The coupling of these two matrices is now complete, and the final matrix is shown in equation A.7. The next matrix to be coupled can now be found and the process repeated.

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad \text{(EQ. A.7.)}$$

In total, twelve entries were filled in, and five questions were asked to the modellers. This equates to 42% of the possible questions being answered directly by the modellers.

Obviously, the bigger the two matrices, the larger the inference opportunities will be. When N_2 is larger (say fifty by fifty), it is a laborious process to raise it to progressively higher powers in the search of Φ. Computers are well suited to large amounts of 'number-crunching', and so are ideally suited to the task. Older computers can take thirty seconds or longer to calculate the Φ matrix and to ask the next question, which means that adequate computing power is a must.

A.3. FROM THE REACHABILITY MATRIX TO THE DIGRAPH

A.3.1. BLOCK ORDERING OF MATRICES

Once the modelling process is complete, but before a matrix can be displayed as a digraph, it goes through a process called block ordering. The elements are grouped in different blocks so that is known at what level or stage they fit into the digraph. There are two methods for block ordering a matrix - either by stages or by levels.

A.3.1.1. BLOCK ORDERING BY LEVELS

Let $M(X,X)$ be a reachability matrix. This matrix can be partitioned into l blocks, called *levels* as follows :

Let $\pi_l(X) = L1,L2....Ll$ where :

$L_i = \{x_i \in (X-L_0-L_1-...L_{i-1}) :$

$$S_{i-1}(x_i) = S_{i-1}(x_i) \cap A_{i-1}(x_i)\}$$

and : $L_0 = \phi$.

$S_{i-1}(x_i)$ is the subset of $X-L_0-L_1-...-L_{i-1}$ that is succedent to x_i.

$A_{i-1}(x_i)$ is the subset of $X-L_0-L_1-...-L_{i-1}$ that is antecedent to x_i.

The process is an iterative one, beginning with L_1. The sets $A(_{i-1})$ and $S(_{i-1})$ can easily be found from the rows and columns of M. If M is as shown in equation A.8., then equation A.9. is the same matrix block ordered by levels. As can be seen that the matrix is always lower triangular, and that each sub-matrix on the main diagonal is symmetric.

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \qquad \text{(EQ. A.8.)}$$

BLOCK ORDER BY LEVELS $M =$

1	0	0	0	0	0	0	0	0	
0	1	0	0	0	0	0	0	0	Level 1
0	0	1	0	0	0	0	0	0	
1	1	0	1	0	0	0	0	0	Level 2
1	1	0	1	1	0	0	0	0	Level 3
1	1	1	1	0	1	0	0	0	
1	1	1	1	1	0	1	0	0	Level 4
1	1	1	1	1	1	1	1	0	Level 5
1	1	1	1	1	1	1	1	1	Level 6

(EQ. A.9.)

BLOCK ORDER BY STAGES $M =$

1	0	1	0	1	1	1	1	1	
0	1	1	0	1	1	1	1	1	Stage 1
0	0	1	0	1	1	1	1	1	Stage 2
0	0	0	1	0	1	1	1	1	Stage 3
0	0	0	0	1	1	0	1	1	
0	0	0	0	0	1	0	1	1	Stage 4
0	0	0	0	0	0	1	1	1	
0	0	0	0	0	0	0	1	1	Stage 5
0	0	0	0	0	0	0	0	1	Stage 6

(EQ. A.10.)

A.3.1.2. BLOCK ORDERING BY STAGES

Let $M(X,X)$ be a reachability matrix. This matrix can be partitioned into l blocks, called *stages* as follows :

Let $\pi_l(X) = G1,G2....Gl$ where :

$G_j = \{x_j \in (X-G_0-G_1-...G_{j-1}) :$

$$A_{j-1}(x_j) = A_{j-1}(x_j) \cap S_{j-1}(x_j)\}$$

and : $L_0 = \phi$.

$S_{j-1}(x_i)$ is the subset of $X-G_0-G_1-...-G_{j-1}$ that is succedent to x_j.

$A_{j-1}(x_i)$ is the subset of $X-G_0-G_1-...-G_{j-1}$ that is antecedent to x_j.

The process is an iterative one, beginning with G_1. The sets $A(_{j-1})$ and $S(_{j-1})$ can easily be found from the rows and columns of M. If M is as shown in equation A.8., then equation A.10. is the same matrix block ordered by stages. As can be seen that the matrix is always upper triangular, and must be transposed to make it lower triangular. Also, as when block ordering by levels, the submatrices on the main diagonals are symmetric.

A.3.2. CONDENSATION MATRIX

Once the matrix has been block ordered by levels or by stages, it is necessary to condense the matrix before processing it any further. This is known as finding the *condensation matrix*.

Let $M(E,E)$ be the a reachability matrix. Let P be a set of proxy elements formed from E as follows :

A. Each element not part of a cycle set will be its own proxy.
B. Each maximal cycle set (feedback set) will have one element as its proxy element.
C. A matrix C can be formed from M for the proxy set P by requiring that if any element of E is antecedent to another element of E, the proxy elements must be in the same relation.

The matrix C is known as the condensation matrix of M.

A.3.3. SKELETON MATRIX

Once a block ordered condensation matrix has been found, one step remains before the digraph can be plotted. Considering the condensation matrix shown in equation A.11., indexed by the elements {A,B,C,D} :

A '1' in the matrix C shows that the two elements have a relationship between them, while a '0' indicates that there is no relationship between them. So, in plotting a digraph, all that is needed is to place the elements in their respective positions in accordance with their levels, and draw an arrow between the elements wherever there is a relationship. The digraph of matrix A.11. is shown in figure A.3.

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \qquad \text{(EQ. A.11.)}$$

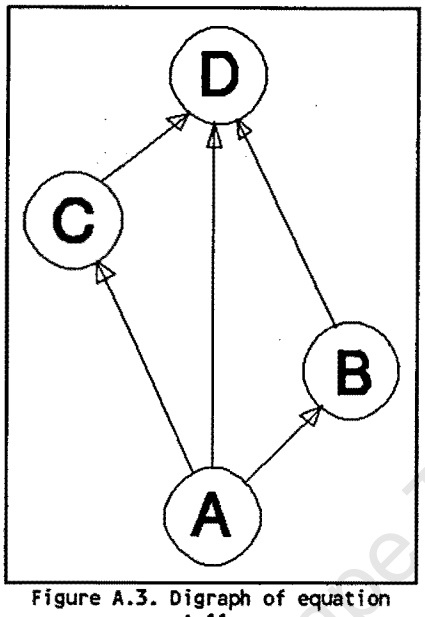


Figure A.3. Digraph of equation A.11.

As can be seen from figure A.3., this digraph has superfluous information. A enhances B, and B enhances D. Thus by implication A enhances D and there is no need for the arrow from A to D. Thus further processing of the matrix is needed before a digraph showing the minimum number of edges can be displayed. (An *edge* is a 1 entry in the matrix, and is shown as an arrow in the digraph.) This process is known as finding the *skeleton matrix* of *C*. It is called the *skeleton matrix* as it the matrix with the minimum number of entries from which *C* can be recreated.

In general, there exists a set of matrices for which *C* is the transitive closure. Among this set there is a reflexive member matrix *K* having the property that if any 1 not lying on the main diagonal is changed to a 0, the resulting matrix will not have *C* as its transitive closure. The matrix *K'* found by subtracting *I* from *K* is called the

skeleton matrix of C.

For any matrix M of dimension n, it is possible to expand M into a sum of its diagonal components of the form :

$$M = \sum_{-n}^{n} M(d)$$

Similarly C and K can be written in that form. To find the skeleton matrix, the following procedure is followed :

A. Let $B(0) = C(0)$.
B. Let $B(-1) = C(-1)$.
C. Set d to 1.
D. Find $B = \sum_{0}^{-d} B(-i)$
E. Find $E = C - C^{n-1}$
F. If $E = \phi$ then stop the process
G. $B(-d-1) = E(-d-1)$
H. Add one to d.
I. Repeat process from step D.

In plotting the digraph, each element is placed on it's respective level, and an arrow is drawn between the elements where a '1' is shown in the matrix. Whenever a proxy element is encountered, the relevant elements are inserted in the digraph, and outlined to show that they are in a feedback set.

Thus for matrix A.11., the matrices $C(0)$ to $C(-3)$ are shown in equations A.12.:

$$C(0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad C(-1) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$C(-2) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \qquad C(-3) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \text{(EQ.A.12.)}$$

From equation A.12., B can be found to be :

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

From step E, the matrix E can be found :

$$E = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

From step G, $B(-2) = E(-2)$, thus:

$$B(-2) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

So, the new matrix B is :

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

Thus the new E = ϕ, and so the skeleton matrix of B11 is :

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

The digraph of this matrix is shown in figure A.4., and as can be seen, there is no superfluous information.

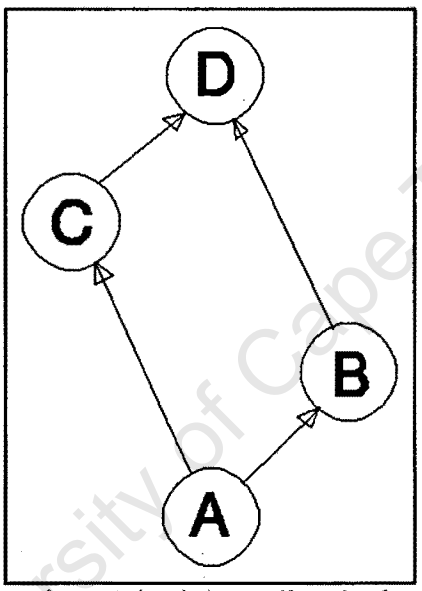


Figure A.4. Skeleton digraph of equation A.11.

APPENDIX B

WRITING THE ISM PROGRAM

B.1. THE CHOICE OF LANGUAGE

The program was developed in TURBO-PASCAL version 7. Upon examination of the language, prior to starting programming, it was seen as a very easy language in which to program. A large amount of literature was available on Pascal, and the language itself is widely used. On this basis the language was chosen.

However, the following factors were not properly examined :

B.1.1. LACK OF BUILT-IN MATRIX FUNCTIONS

Pascal is not ideally suited for matrix manipulations as is has no 'built-in' matrix manipulation routines, and so all routines for matrix manipulations had to be written. Though these are not in themselves difficult routines to write, pascal compounds this problem with another drawback :

Pascal is strongly typed. That means that every variable (and thus every matrix) has to be given a type when it is defined. So, while the reachability matrix (M) required for ISM was of one type, the Φ matrix required for the coupling method is of another type, as it is required to be much larger. This results in the duplication of routines to cater for the different types of matrices.

B.1.2. PASCAL'S MEMORY LIMITATIONS

Pascal has a limit of 64 Kb of memory for variables. So, if the Φ matrix was given the dimensions 150 by 150, it would require 22.5 Kb of memory. In order to form the Φ matrix, the matrix $N2$ has to also be in memory, which would mean 45 Kb of free memory would be used for these matrices alone! This would in turn leave very little room for other variables such as the description of the elements.

This second factor proved to be the major bugbear in

programming, primarily because of a failure to realise how large the matrices become in ISM, and how small 64 Kb is to store these matrices. It would have been more desirable to write the program in 'Fortran' or 'C' in order to have avoided both these problems.

However, the most positive spin-off of having written the program in Pascal, is that is can run on any IBM PC compatible with 1 Mb of RAM and a 286 microprocessor. This means that it can run on any IBM PC compatible computer that can be bought today.

B.2. THE STRUCTURE OF THE PROGRAM

The structure of the program is shown in figure B.1. As can be seen there is a main control program, and eight units. A brief description of each unit is given below :

Main is the main control program and only runs the menu. From it the units are called depending on the choice of the operator.

Typeunit is common to all other units, but the links are not shown in figure B.1. for the sake of clarity. In it are the definitions of common variable types, as well as all common procedures.

Enterunit is the unit that allows the operator to enter the contextual relation and the description of the elements that are to be modelled.

Contunit is the control unit for the modelling process. It looks at the matrix *M*, and depending on the state of the matrix decides whether to proceed to *thiunit* or *passunit*.

Passunit controls the scanning method, and partitions the

matrix on an element.

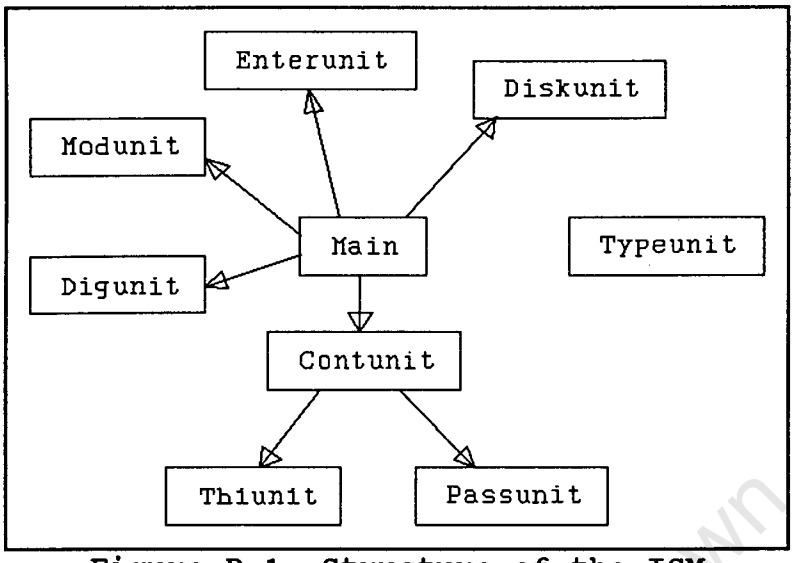


Figure B.1. Structure of the ISM program.

Thiunit is used to couple the sub-matrices together once *M* has been partitioned on elements.

Digunit plots the digraphs of the completed models on the screen.

Modunit allows the operator to modify the model by cutting or adding links, and by removing or adding elements.

Diskunit saves or retrieves the model with its element descriptions.

Due to memory limitations, a maximum of 50 elements can be modelled - more than enough for most applications. The program has a matrix *MREACH* that contains all information relevant to the model. (*MREACH* is the equivalent to the matrix *M* in appendix A - The mathematics of ISM.) This matrix is in fact 51 by 51, as it ranges from 0 to 50. In the zero column and row, the element number is stored. This element number gives

the index of the element in the array of element descriptions. The rest of the matrix contain the 0's and 1's that make up the model.

B.3. PROGRAMMING THE MATHEMATICS

The mathematics of ISM is not complicated but cumbersome. Matrix manipulations are easy, but the size of them, combined with the memory limitations of Pascal make the task more daunting. All the problems encountered in programming related to Pascal's memory limitations, and not to the mathematics involved.

The only way to surpass this memory problem was to limit the global variables to a minimum, and to use local variables within each unit. The memory limits were most restrictive in passunit and in thiunit, the latter giving the greater problems.

These two units gave the most problems for the following reasons :

1. They involve the most matrix manipulation.

2. They were the first two units to be written, and as better programming skills were developed, further problems were avoided.

What follows is a brief description of how the problems within passunit and thiunit were overcome :

B.3.1. OVERCOMING PROBLEMS IN PASSUNIT

Passunit is the unit that partitions the matrix on an element. The process of partitioning by elements is an iterative one. Warfield in his approach suggests that for every iteration, the lift, vacancy and drop sets be

repeatedly broken into smaller and smaller submatrices, until all submatrices are zero or one in size. (See figure A.2. in appendix A).

However, a memory problem arises in that each submatrix is of an unknown size until the modeller has answered the questions. So, in programming for a fifty element ISM, one must allow for the lift set contain up to 49 elements after the first pass. After the second pass, the lift set may contain 48 elements, repeating itself for a possible 48th pass, that would leave one element in the lift set. However, this unlikely scenario may happen in the drop or vacancy set instead of the lift set.

This can result in a potentially monumental amount of memory to be needed for the partitioning on elements. [Remember, each matrix must be given a type, and if for each pass a different matrix type is used to allow for the reduction in size, the code must be re-written to allow for this.]

So, instead of storing all the submatrices, only the start an end points of the various sets were stored and the *MREACH* matrix was re-ordered after each pass. This vastly reduced memory requirements, as the *MREACH* matrix is the only matrix necessary.

This can however still result in some programming problems. For every iteration a progressively larger and larger array is needed to store all the data of the start and stop points of the sub-matrices. By the end of cycle 2, 53 different matrices need to be recorded.

So, the program was written to concentrate on one set, and to keep iterating that set before starting on the next set. With reference to figure A.2., the following steps are observed :

1. The full set of cycle one is found.
2. The output for the next cycle is found as follows :

   a. If the lift set from the current cycle contains more than one entry, it is partitioned further. Steps 2a and 2b are ignored, and the output is processed from step 2.
   b. If the lift set is found to contain one or less entries, and the vacancy set has more than one entry, the vacancy set is partitioned further to give the output for the next cycle. Step 2c is ignored, and the output is processed from step 2.
   c. If both the lift set and vacancy set contain one or less entries, and the drop set has more than one entry, the drop set is partitioned to give the output from the next cycle. The cycle is repeated from step 2.

3. If all matrices for the current cycle are found to be of size one or zero, the previous cycle is reprocessed from step 2.

Thus if after cycle 1 the lift set has two entries, and the drop set 4, the lift set would be partitioned further. Once this was done, all the output sets would be size one or zero. So, the process would return to the first cycle, and the drop set would then be partitioned further.

This results in a minimum number of sets having to be stored in memory at any point in time, and so frees up memory. However, it does result in certain elements appearing in questions very frequently for a while, and then seemingly disappearing from the modelling process.

The real solution to this problem is to use pointers. This is because pointers in pascal do not use the 64Kb memory

assigned to variables, but the rest of the computer's free memory. This alleviates the memory problem. Each pointer should store the start and end positions of the set it represents in *MREACH*, and also point to the address of the lift, drop and vacancy sets that were formed from it. These sets each in turn store their start and end points within *MREACH*, and the lift drop and vacancy sets formed from them. This process would repeat itself until all sets are of size one or zero.

B.3.2. OVERCOMING PROBLEMS IN THIUNIT

Thiunit is the unit that couples matrices together. It is given the start and end points of matrices D and E (as mentioned in appendix A) and from them forms the Φ matrix to imply answers to the questions.

The problem in thiunit was also one of memory. As was mentioned earlier a 150 by 150 matrix uses 22.5 Kb of the available 64 Kb. This memory problem meant that Φ had to be limited in size to 100 by 100 for the program to run. This is done quite simply by reducing the size of the E matrix until 2*(size of D)*(size of E) was equal to or less than 100.

This restriction gave rise to the following problem, that took a several weeks to find, but very little time to solve. With reference to figure B.2., let all entries in the matrices D_1 and E are known, and that all entries not in the matrices are unknown, then if D_1 is 10 by 10, and E is 6 by 6, the matrix Φ will be 120 by 120.

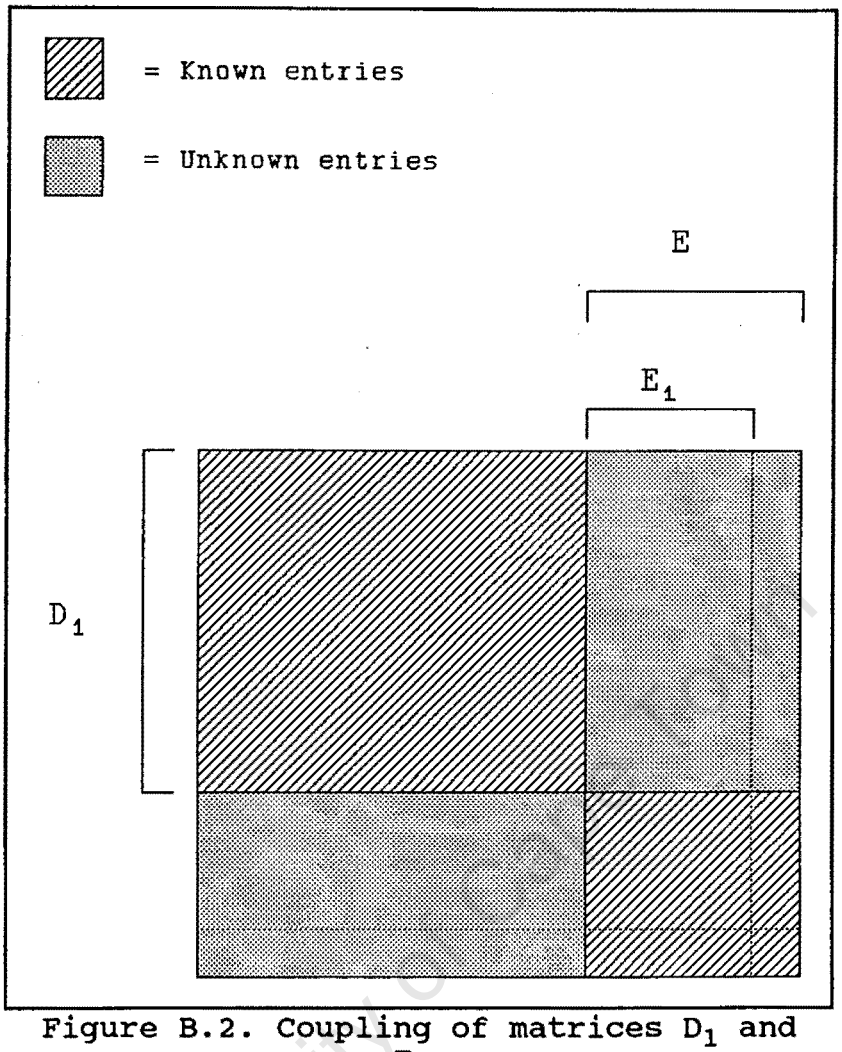


Figure B.2. Coupling of matrices D_1 and E_1.

However, the program will limit E, and find E_1 of size 5 by 5 by discarding the last element. Modelling will then proceed for the Φ matrix. The program will then find a new matrix D_2 of size 15 by 15 consisting of the old matrices D_1 and E_1 plus the answers found during modelling. A new matrix E_2 of size 1 by 1 will also be found that consists of the previously discarded element. As can be seen from figure B.3., not all the entries outside of D_2 and E_2 are unknown. The submatrices K_1 and K_2 in the figure show the known entries in the matrix.

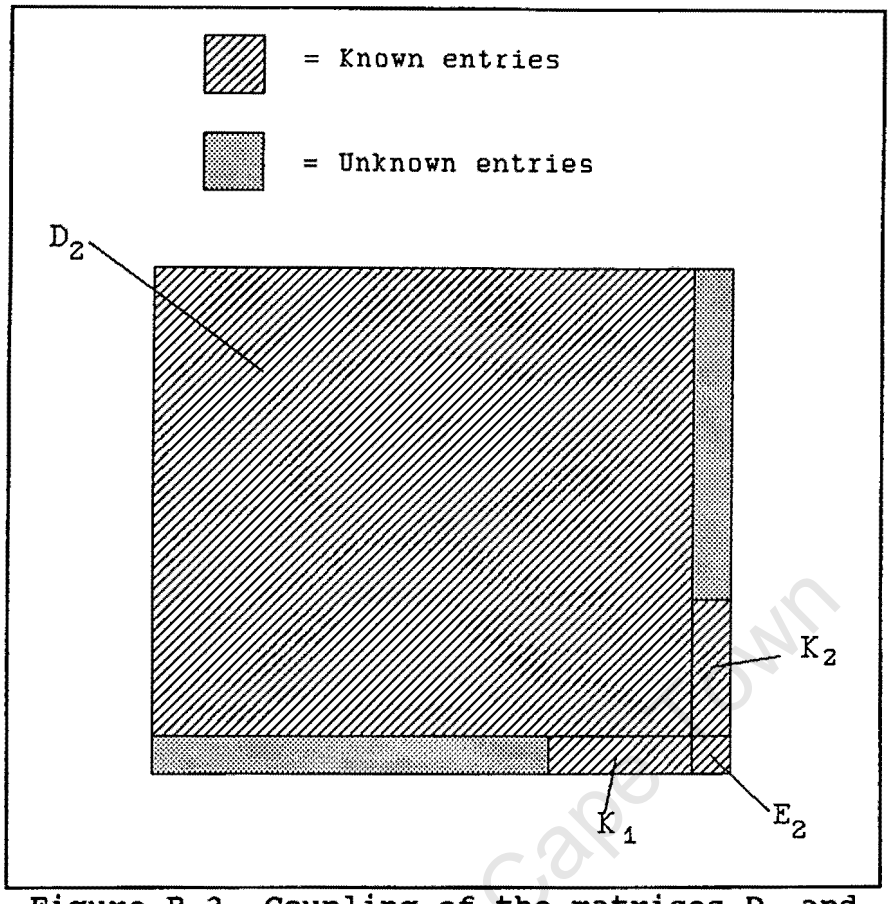


Figure B.3. Coupling of the matrices D_2 and E_2.

The program initially sought new answers to these known entries. All that was needed was to delete the elements of the Φ matrix that represent entries already known in M, before going on to ask the questions that reveal the unknown entries.

This fault was particularly frustrating for two reasons :

1. Even though it does not appear so, it was a memory problem. If enough memory had been available, the Φ matrix would not have been reduced in size, and the fault would not have occurred.

2. The fault was an intermittent one - it only occurred when the size of Φ had been reduced, and this did not

occur that frequently. This made it particularly difficult to trace, as all manipulations made to form the Φ matrix were correct, all the answers implied from the Φ matrix were correctly implied, and the correct entries were deleted from the matrix as the implications were made. However, the output did not agree with the data fed into the computer.

B.4. DEBUGGING AND BENCHMARKING

Programs involving matrix manipulations are particularly difficult to debug, especially when the matrices are large. This is because manipulations of large matrices are tedious to do by hand, and errors easily creep in. So, in order to benchmark the program and ensure that it had no programming errors, worked examples of matrix manipulations from Warfield[12] were used as input to the program, and the results of the program were compared with those of the book until all the bugs were removed.

Once all it was found that the program was giving the correct results, models shown in various papers on ISM were fed into the program, and further de-bugging proceeded until the results were identical.

Also, 'home-made' models (i.e. any hand-drawn selection of elements and links) were fed into the program, and came out the same as the original. The element numbers were then swopped around with the layout remaining the same, and still the correct answer was given by the program.

As a final test, the program was benchmarked against the ISM package of Ross Janes, and found to be perfect. The program was then pronounced to be 'error-free'.

APPENDIX C

LISTING OF THE ISM PROGRAM

MAIN

```
Program Main;
{$M 65520,0,655360}
{Main program to enter and exit units as needed to enter elements, model
elements, edit elements and all the rest.}

uses
    crt,enterunit,modunit,contunit,typeunit,digunit,diskunit;
var
   MREACH                          : tarr51;
   ELEMENT                         : tars51;
   FEEDCNTR,MRSIZE,POSITION,I,J    : shortint;
   CONTEXT,RELATE                  : string;
   FEEDARR                         : tar51;
   JUNK,JUNK1                      : char;
   BQ                              : tar3;
   BN                              : tar2;

Procedure backg;
{Input background knowledge of the person being interviewed.}
begin
 clrscr;
 Write('Have you done a formal tertiary course in quality control? ');
 BQ[1] := readyn;
 clrscr;
 write('Have you done an in-house course/seminar on quality control? ');
 BQ[2] := readyn;
 clrscr;
 writeln('have you read any books by Philip Crosby, Joseph Juran, W. Edwards
         Deming ');
 write('or Kaoru Ishikawa? ');
 BQ[3] := readyn;
 clrscr;
 write('Do you actively support your company''s quality program? ');
 BQ[4] := readyn;
 clrscr;
 write('How do you rate your quality program (1 = poor, 10 = excellent) ');
 readln(BN[1]);
 clrscr;
 write('How many years have you been involved in a quality program? ');
 readln(BN[2]);
 clrscr;
 write('Approximately how many books/articles on quality control have you read
       in the last two years?');
 readln(BN[3]);
 clrscr;
end;

Procedure Init;
{Procedure to initialise variables upon starting the program.}
begin
 For I := 0 to 50 do begin
     for J := 0 to 50 do begin
         MREACH[I,J] := 2;
     end;
     ELEMENT[I] := '';
     FEEDARR[I] := 0;
 end;
 JUNK := 'd';
 FEEDCNTR := 0;
 MRSIZE := 0;
 POSITION := 0;
 CONTEXT := '';
 RELATE := '';
end;
```

MAIN

```
{Main program}
begin
 init;
 While JUNK <> 'X' do begin
 clrscr;
 Writeln('Please enter the letter corresponding to your choice:');
 Writeln;
 Writeln('(A) : Enter elements for a new model.');
 Writeln('(B) : Model elements to form reachability matrix.');
 Writeln('(C) : Modify (add/delete) elements or links.');
 Writeln('(D) : Plot digraph of elements.');
 Writeln('(E) : Save or Restore model from disk.');
 Writeln('(F) : Determine background knowledge.');
 Writeln;
 writeln('(X) : EXIT.');
 Writeln;
 Writeln('Please enter ''A'',''B'',''C'',''D'',''E'', or ''X''. ');
 JUNK := readkey;
 JUNK := Upcase(JUNK);
 dellay;
 Case JUNK of
     'A' : begin
                if MRSIZE > 0 then begin
                   clrscr;
                   writeln('Entering new elements will erase all current
elements.');
                   writeln('If you wish to add, delete or modify current
elements ');
                   writeln('please choose option (C) Modify elements or
links.');
                   Writeln;
                   write('Are you sure that you wish to erase current elements
                      (Y/N)?');
                   JUNK1 := readyn;
                   dellay;
                   case junk1 of
                   'Y'     : begin
                             init;
                             elenter(ELEMENT,CONTEXT,RELATE,MRSIZE,MREACH);
                             end;
                   end;
                end;
                if mrsize = 0 then begin
                 ELENTER(ELEMENT,CONTEXT,RELATE,MRSIZE,MREACH);
              end;
           end;
     'B': control(MREACH,ELEMENT,CONTEXT,RELATE,FEEDARR,MRSIZE,FEEDCNTR,
                POSITION);
     'C' : modify(MREACH,ELEMENT,FEEDARR,FEEDCNTR,MRSIZE);
     'D' : digraph(MREACH,ELEMENT,FEEDARR,FEEDCNTR,MRSIZE);
     'E':disk(MREACH,ELEMENT,FEEDARR,FEEDCNTR,MRSIZE,CONTEXT,RELATE,BQ,BN);
     'F' : backg;
 end;
end;
end.
```

TYPEUNIT

```
unit typeunit;

interface
 uses
    crt;

 type
    tar51  = array[0..50] of shortint;
    tarr51 = array[0..50,0..50] of shortint;
    tarr1  = array[0..6,1..2] of shortint;
    tars51 = array[1..50] of string;
    tar2   = array[1..3] of shortint;
    tar3   = array[1..4] of char;

 Function readyn : char;
 Procedure wait;
 Procedure thinking;
 Procedure dellay;
 Procedure matexpand(var MREACH : tarr51; var feedarr : tar51;
                     var MRSIZE,feedcntr : shortint);
 Procedure matreduce(var MREACH : tarr51; var feedarr : tar51;
                     var MRSIZE,feedcntr : shortint);

implementation

Function readyn : char;
var
   JUNKS : char;
   QQ    : shortint;
begin
 JUNKS := ' ';
 QQ := 0;
 while QQ <> 1 do begin
    JUNKS := readkey;
    JUNKS := upcase(JUNKS);
    if JUNKS <> 'Y' then QQ := 1;
    if JUNKS <> 'N' then QQ := 1;
 end;
 readyn := JUNKS;
end;

Procedure thinking;
begin
  write('.');
end;

Procedure wait;
begin
 write('.');
end;

procedure dellay;
begin
 delay(200);
 while keypressed do begin
       readkey;
       dellay;
 end;
end;
```

```
Procedure matexpand(var MREACH : tarr51; var feedarr : tar51;
                    var MRSIZE,feedcntr : shortint);
{Procedure to expand the matrix to include feedback sets - for modunit and
diskunit.}
var
   MTEMP                          : TARR51;
   CHECKER,I,J,K,L,M,N,P,Q        : shortint;
begin
 Checker := 1;
 M := 2;
 N := 1;
 While Checker < FEEDCNTR do begin
       L := 0;
       I := 1;
       While I <= MRSIZE do begin
             If M < FEEDCNTR then begin
                If MREACH[0,I+L] = FEEDARR[M] then begin
                   inc(M);
                   For Q := 1 to (FEEDARR[n]-1) do begin
                       for J := (MRSIZE+L+Q) downto I+L+Q do begin
                           move(MREACH[J-1],MREACH[J],
                                MRSIZE+L+FEEDARR[N]-1);
                       end;
                       for J := (MRSIZE+L+Q) downto I+L+Q do begin
                           for P:= 0 to (MRSIZE+L+Q) do begin
                               MREACH[P,J] := MREACH[P,J-1];
                           end;
                       end;
                       MREACH[0,I+Q+L] := FEEDARR[M];
                       MREACH[I+Q+L,0] := FEEDARR[M];
                       inc(M);
                   end;
                   L := L + (FEEDARR[N]-1);
                   Checker := checker + (FEEDARR[N]+1);
                   N := M;
                   inc(M);
                end;
             end;
             inc(I);
       end;
       MRSIZE := MRSIZE + L;
 end;
end;
```

```
Procedure matreduce(var MREACH : tarr51; var FEEDARR : tar51;
                    var MRSIZE,FEEDCNTR : shortint);
{Procedure to remove the feedback sets from a matrix for modunit and
diskunit.}
var
   MTEMP                          : tarr51;
   CHECKER,I,J,K,L,M,N,P,Q,R      : shortint;
Begin
 I := 1;
 N := 1;
 M := 1;
 FEEDCNTR := 2;
 While I < MRSIZE do begin
     FEEDARR[FEEDCNTR] := MREACH[I,0];
     J := I+1;
     While J <= MRSIZE do begin
         L := 0;
         For K := 1 to MRSIZE do begin
             If MREACH[J,K] = MREACH[I,K] then inc(L);
             If MREACH[K,J] = MREACH[K,I] then inc(L);
         end;
         If L = (2*MRSIZE) then begin
            inc(FEEDCNTR);
            FEEDARR[FEEDCNTR] := MREACH[0,J];
            inc(N);
         end;
         inc(J);
     end;
     If N > 1 then begin
        inc(FEEDCNTR);
        FEEDARR[M] := N;
        For P := 2 to N do begin
            Q := 1;
            while MREACH[Q,0] <> FEEDARR[M+P] do begin
                inc(Q);
            end;
            For R := (Q+1) to MRSIZE do begin
                move(MREACH[R],MREACH[R-1],MRSIZE+1);
                for K := 0 to MRSIZE do begin
                    MREACH[K,R-1] := MREACH[K,R];
                end;
            end;
            for R:= 1 to MRSIZE do begin
                MREACH[R,MRSIZE] := 0;
                MREACH[MRSIZE,R] := 0;
            end;
            dec(MRSIZE);
        end;
        M := M + N + 1;
        N := 1;
        inc(FEEDCNTR);
     end;
     inc(I);
 end;
 FEEDARR[FEEDCNTR]:=0;
 FEEDCNTR := FEEDCNTR-2;
end;
end.
```

ENTERUNIT

```
Unit ENTERUNIT;
{Unit to enter element data, the context and the relationship between
element for ISM.}
Interface
 uses
   crt,typeunit;

 Procedure elenter(var ELEMENT : tars51; var CONTEXT,RELATE : String;
                   var MRSIZE : shortint; var MREACH : TARR51);

Implementation
var
   I        : shortint;

Procedure elenter;
begin
 I := 0;
 Clrscr;
 Writeln ('Please enter the context of the elements :');
 writeln;
 Readln (CONTEXT);
 writeln;
 writeln;
 writeln ('Please enter the relationship between the elements :');
 writeln;
 Readln (RELATE);
 while I < 50  do begin
       Clrscr;
       Writeln('Please enter the description of the elements to be modeled
              :');
       Writeln('(Enter ''X'' to exit)');
       inc(I);
       Writeln;
       Writeln('Please enter element ',I);
       Writeln;
       Readln(ELEMENT[I]);
       if ELEMENT[I] = 'x' then begin
          MRSIZE := I-1;
          I := 51;
       end;
       if ELEMENT[I] = 'X' then begin
          MRSIZE := I-1;
          I := 51;
       end;
 end;
 for I := 1 to MRSIZE do begin
      MREACH[I,0] := I;
      MREACH[0,I] := I;
      MREACH[I,I] := 1;
 end;
end;
end.
```

CONTUNIT

```
unit contunit;
{Program to ensure that when matrix A&B are interconnected that best
use can be made of the thispecial matrix. At the same time, it can
be ensured that thi stays within the limit of 100 elements. Also ensures
that the matrix is first partitioned on an element before adding the
matrices together.}

Interface

uses
    crt,passunit,thiunit,typeunit;

  Procedure control(var MREACH : tarr51; var ELEMENT : TARS51;
                    var CONTEXT,RELATE : STRING; var FEEDARR : TAR51;
                    var MRSIZE,FEEDCNTR,POSITION : SHORTINT);

Implementation

Var
   I,J,COUNTER,SIZE              : shortint;
   MARKER,POINT,SPECIAL          : shortint;
   SIZEA,SIZEB,EVENT             : shortint;
   BOOL                          : char;

Procedure findmat(var POINT,MRSIZE,MARKER : shortint; MREACH : TARR51);
{Procedure to find the start and end of A and B matrices so that modelling can
begin}
begin
 I:= MARKER + 1;
 POINT := MRSIZE;
 while I <= POINT do begin
       J := MARKER + 1;
       while J <= POINT do begin
             if MREACH[I,J] = 2 then begin
                if I>J then begin
                POINT := I - 1;
                end else begin;
                POINT := J-1;
                end;
             end;
             inc(J);
       end;
       inc(I);
 end;
end;

Procedure checkint(MREACH : TARR51);
{Procedure to see if thispecial can be used}
begin
 SPECIAL := 0;
 for I := 1 to SIZEA do begin
     for J := 1 to SIZEB do begin
         if MREACH[I,SIZEA+J] = 2 then inc(SPECIAL);
     end;
 end;
end;
```

```
Procedure findab(var POSITION,SIZEA,SIZEB : shortint; MREACH : tarr51;
                 MRSIZE : shortint);
{Procedure to control the finding of the A & B matrices.}
begin
 EVENT := 0;
 MARKER := 0;
 findmat(POINT,MRSIZE,MARKER,MREACH);
 SIZEA := POINT;
 if POINT = MRSIZE then begin
{mark that modelling is complete}
   EVENT := 2;
 end else begin
{count 1 off diagonal to see if rest of matrix has been through passone}
   J := 0;
   for I := POINT to MRSIZE-1 do begin
       J := J + MREACH[I+1,I];
   end;
{if everything below has to be modelled, then if point is one, position = 0,
otherwise position is point to model the last few (added) elements only. If
there is only one added element, then the thi matrix must be found.}
   if J = (2*(MRSIZE-POINT)) then begin
      if (MRSIZE-POINT) > 1 then begin
         EVENT := 1;
         if POINT = 1 then POSITION := 0;
         if POINT > 1 then POSITION := POINT;
      end else begin
          EVENT := 0;
          SIZEB := 1;
      end;
   end else begin
      MARKER := SIZEA;
      findmat(POINT,MRSIZE,MARKER,MREACH);
      SIZEB := (POINT-MARKER);
   checkint(MREACH);
   end;
 end;
end;

Procedure control(var MREACH : tarr51; var ELEMENT : TARS51;
                  var CONTEXT,RELATE : STRING; var FEEDARR : TAR51;
                  var MRSIZE,FEEDCNTR,POSITION : SHORTINT);
{Main procedure to control all this - ie whether partitioning on an element
or adding on the extra matrix.}
begin
 EVENT := 0;
 While EVENT <> 2 do begin
       findab(POSITION,SIZEA,SIZEB,MREACH,MRSIZE);
       Case EVENT of
            0: begin
                SPECIAL := 1;
                matexpand(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
                findab(POSITION,SIZEA,SIZEB,MREACH,MRSIZE);
                thifind(MREACH,MRSIZE,SIZEA,SIZEB,SPECIAL,CONTEXT,RELATE,
                     ELEMENT);
                matreduce(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
               end;
            1: begin
                passone(MREACH,FEEDARR,FEEDCNTR,POSITION,MRSIZE,ELEMENT,
                     CONTEXT,RELATE);
               end;
       end;
 end;
end;
end.
```

THIUNIT

```
Unit THIUNIT;
{program of various routines designed to manipulate sets.
eg finding transpose, inverse(copmplement), multiplication
and boolean addition. Also finds THI matrix, finds and asks the best
questions to minimise the number of questions.}

Interface
 uses
     crt,typeunit;

 type
     tarr5   = array[1..30,1..30] of shortint;
     tarr100 = array[0..100,0..100] of shortint;
     tar100  = array[0..100] of integer;

Procedure thifind(var MREACH : tarr51; var MRSIZE : shortint;
                  SIZEA,SIZEB,special : shortint; CONTEXT,RELATE : string;
                  ELEMENT : tars51);

Implementation

var
   A,MREACHT                    : tarr51;
   B,N1                         : tarr5;
   MRSIZET                      : shortint;
   I,J,K                        : shortint;
   SIZEAT, SIZEBT               : shortint;
   N2SIZE,N1SIZE,L,M,N,MARK,RCR    : integer;

Procedure suan(var SU,AN : tar100; THI : tarr100; N2SIZE : integer);
{procedure to find size of antecedent and succedent sets of THI}
begin
 for I := 1 to N2SIZE do begin
     SU[I] := 0;
     AN[I] := 0;
 end;
 for I := 1 to N2SIZE do begin
     for J := 1 to N2SIZE do begin
         if THI[I,J] = 1 then inc(SU[I]);
         if THI[J,I] = 1 then inc(AN[I]);
     end;
     dec(SU[I]);
     dec(AN[I]);
 end;
end;

function min(SU,AN : tar100; I : integer): integer;
{find the smaller of 2 numbers.}
begin
 if SU[I] < AN[I] then begin
    MIN := SU[I];
 end else begin
     MIN := AN[I];
 end;
end;
```

```
Procedure bestchoice(var MARK : integer; SU,AN : tar100; N2SIZE : integer);
{Procedure to find the bestchoice question that will infer the most
answers.}
var
   POS : integer;
   BC  : tar100;
begin
 for I:= 1 to N2SIZE do begin
     BC[I] := min(SU,AN,I);
 end;
 MARK := 1;
 POS := BC[1];
 for I := 2 to N2SIZE do begin
     if BC[I] = POS then begin
        if (SU[I]+AN[I]) >= (SU[MARK]+AN[MARK]) then begin
           POS := BC[I];
           MARK := I;
        end;
     end;
     if BC[I] > POS then begin
        POS := BC[I];
        MARK := I;
     end;
 end;
end;

Procedure dimin(var THI:tarr100; var N2SIZE:integer; RCR : integer);
{proceedure to diminish the size THI as questions are answered}
begin
 for M := (RCR+1) to N2SIZE do begin
     move(THI[M],THI[M-1],N2SIZE+1);
     for N := 0 to (N2SIZE+1) do begin
         THI[N,M-1] := THI[N,M];
     end;
 end;
 dec(N2SIZE);
end;
```

```
Procedure thiyes(var MREACHT : tarr51; var TOBEDEL : tar100;
                 THI : tarr100; MARK, N2SIZE : integer; SIZEAT,I : shortint);
{procedure to full in the original and inferred answers to a yes reply
to the best question from the THI matrix}
begin
 I := 0;
 if THI[MARK,0] > SIZEAT then begin
   for L := 1 to N2SIZE do begin
    if THI[MARK,L] = 1 then begin
       if THI[L,0] > SIZEAT then begin
          MREACHT[THI[L,0],THI[0,L]] := 1;
       end else begin
           MREACHT[THI[L,0],THI[0,L]] := 0;
       end;
      inc(I);
      TOBEDEL[I] := L;
    end;
   end;
 end else begin
     for L := 1 to N2SIZE do begin
         if THI[L,MARK] = 1 then begin
            if THI[L,0] > SIZEAT then begin
               MREACHT[THI[L,0],THI[0,L]] := 0;
            end else begin
                MREACHT[THI[L,0],THI[0,L]] := 1;
            end;
            inc(I);
            TOBEDEL[I] := L;
         end;
     end;
 end;
 TOBEDEL[0] := I;
end;
```

```
Procedure thino(var MREACHT : tarr51; var TOBEDEL : tar100;
                THI : tarr100; MARK, N2SIZE : integer; SIZEAT,I : shortint);
{procedure to fill in the original and inferred answers to a no reply
to the best question from the THI matrix}
begin
 I:=0;
 if THI[MARK,0] > SIZEAT then begin
    for L := 1 to N2SIZE do begin
        if THI[L,MARK] = 1 then begin
           if THI[L,0] > SIZEAT then begin
              MREACHT[THI[L,0],THI[0,L]] := 0;
           end else begin
               MREACHT[THI[L,0],THI[0,L]] := 1;
           end;
           inc(I);
           TOBEDEL[I] := L;
        end;
    end;
 end else begin
    for L := 1 to N2SIZE do begin
        if THI[MARK,L] = 1 then begin
           if THI[L,0] > SIZEAT then begin
              MREACHT[THI[L,0],THI[0,L]] := 1;
           end else begin
              MREACHT[THI[L,0],THI[0,L]] := 0;
           end;
           inc(I);
           TOBEDEL[I] := L;
        end;
    end;
 end;
 TOBEDEL[0] := I;
end;
```

```
Procedure poseques(var MREACHT:tarr51; var THI : tarr100; ELEMENT : tars51;
                   var N2SIZE : integer; MARK,I : integer;
                  CONTEXT,RELATE : string; SIZEAT : shortint);

{Procedure to ask the bestchoice question to the modellers.}
var
   BOOL          : char;
   RCR           : integer;
   TOBEDEL       : tar100;

begin
 clrscr;
 gotoxy(1,5);
 writeln (CONTEXT);
 writeln;
 writeln (ELEMENT[MREACHT[THI[MARK,0],0]]);
 writeln;
 writeln (RELATE);
 writeln;
 writeln (ELEMENT[MREACHT[0,THI[0,MARK]]]);
 writeln;
 write('Y/N ?');
 BOOL :=readyn;
 dellay;
 clrscr;
 case BOOL of
      'Y' : begin
            thiyes(MREACHT,TOBEDEL,THI,MARK,N2SIZE,SIZEAT,I);
      end;
      'N' : begin
            thino(MREACHT,TOBEDEL,THI,MARK,N2SIZE,SIZEAT,I);
      end;
 end;
 for K := tobedel[0] downto 1 do begin
     RCR := TOBEDEL[K];
     dimin(THI,N2SIZE,RCR);
 end;
end;

Procedure fillthi(var THI : tarr100; N2SIZE : integer; ELEMENT : tars51;
                  CONTEXT,RELATE : string);
{Procedure to fill the THI matrix from the point of generation until
all entries are complete.}
var
   SU,AN : tar100;

begin
 while N2SIZE > 0 do begin
       suan(SU,AN,THI,N2SIZE);
       bestchoice(MARK,SU,AN,N2SIZE);
       poseques(MREACHT,THI,ELEMENT,N2SIZE,MARK,I,CONTEXT,RELATE,SIZEAT);
 end;
end;
```

```
procedure complA(var ABAR,ATRANS: tarr51; MRSIZET:shortint);
{Procedure to find complement of a matrix.}
begin
 for I := 1 to MRSIZET do begin
     for J := 1 to MRSIZET do begin
         if ATRANS[I,J] = 1 then begin
            ABAR[I,J] := 0;
         end else begin
             ABAR[I,J] := 1;
         end;
     end;
 end;
end;

procedure complementB(var B,BBAR: tarr5; MRSIZET:shortint);
{Procedure to find complement of a matrix.}
begin
 for I := 1 to MRSIZET do begin
     for J := 1 to MRSIZET do begin
         if B[I,J] = 1 then begin
            BBAR[I,J] := 0;
         end else begin
             BBAR[I,J] := 1;
         end;
     end;
 end;
end;

procedure transpA(var A,ATRANS: tarr51; MRSIZET:shortint);
{Procedure to find transpose of matrix}
begin
 for I := 1 to MRSIZET do begin
     for J := 1 to MRSIZET do begin
         ATRANS[I,J] := A[J,I];
     end;
 end;
end;

procedure transposeB(var B,BTRANS: tarr5; MRSIZET:shortint);
{Procedure to find transpose of matrix}
begin
 for I := 1 to MRSIZET do begin
     for J := 1 to MRSIZET do begin
         BTRANS[I,J] := B[J,I];
     end;
 end;
end;

procedure booladd(var A,B: tarr51; MRSIZET:shortint);
{Procedure to boolean add two matrices.}
begin
 for I := 1 to MRSIZET do begin
     for J := 1 to MRSIZET do begin
         A[I,J] := A[I,J] + B[I,J];
         if A[I,J] > 1 then A[I,J] := 1;
     end;
 end;
end;
```

```
procedure booladdb(var BBAR : tarr5; MRSIZET:shortint);
{Procedure to boolean add two matrices.}
begin
 for I := 1 to MRSIZET do begin
     BBAR[I,I] := 1;
 end;
end;

procedure multiply2(var THI,N2: tarr100; N2SIZE:integer);
{Procedure to multiply two matrices}
var
   TEMP2 : tarr100;

begin
 for I:= 1 to N2SIZE do begin
     for J := 1 to N2SIZE do begin
         THI[I,J] := 0;
         for K := 1 to N2SIZE do begin
             if THI[I,J] = 1 then break;
             THI[I,J] := N2[I,K]*N2[k,J];
         end;
     end;
     thinking;
 end;
end;

Procedure makei(var II: tarr51);
{Procedure to make matrix I of relevant size}
begin
 for I := 1 to 50 do begin
     II[I,I] := 1;
 end;
end;

Procedure samer(var N2,THI : tarr100; N2SIZE : integer);
{Procedure to make two matrices identical.}
begin
 for I := 1 to N2SIZE do begin
     move(THI[I],N2[I],N2SIZE+1);
 end;
end;
```

```
Procedure findthi(var THI : tarr100; var N2SIZE : integer;
                  SIZEAT,SIZEBT : shortint);
{Procedure to find THI by finding the transitive closure of N2.}
var
   N2           : tarr100;
   TOBEDEL      : tar100;
   MTEMP        : tarr51;
   CHECKER      : shortint;

begin
 L := 0;
 while M <> N2SIZE do begin
     thinking;
     M := 0;
     L := 0;
     samer(N2,THI,N2SIZE);
     multiply2(THI,N2,N2SIZE);
     thinking;
     for I := 1 to N2SIZE do begin
         for J := 1 to N2SIZE do begin
             if THI[I,J] = N2[I,J] then inc(L);
         end;
         if L <> N2SIZE then break;
         inc(M);
         L := 0;
     end;
 end;
{fill in x and y reference of z index for THI}
 L := 0;
 K := 0;
 for I := 1 to SIZEBT do begin
     for J := 1 to SIZEAT do begin
         inc(K);
         THI[0,K] := (SIZEAT+I);
         THI[K,0] := J;
     end;
 end;
 for I := 1 to SIZEBT do begin
     for J := 1 to SIZEAT do begin
         inc(K);
         THI[0,K] := J;
         THI[K,0] := (SIZEAT+I);
     end;
 end;
 for I := 0 to MRSIZET do begin
     move(MREACHT[I],MTEMP[I],MRSIZET+1);
 end;
 CHECKER := 1;
{Fill in the THI matrix for the X's specified in previous modelling}
 while CHECKER <> 0 do begin
       CHECKER := 0;
       MARK := 0;
       I := 1;
       while I <= SIZEBT  do begin
             J := 1;
             while J <= SIZEAT do begin
                   inc(MARK);
                   if MTEMP[J,SIZEAT+I] = 1 then begin
                      MTEMP[J,SIZEAT+I] := 3;
                      thiyes(MREACHT,TOBEDEL,THI,MARK,N2SIZE,SIZEAT,I);
                      inc(CHECKER);
                   end else if MTEMP[J,SIZEAT+I] = 0 then begin
                      MTEMP[J,SIZEAT+I] := 3;
                      thino(MREACHT,TOBEDEL,THI,MARK,N2SIZE,SIZEAT,I);
```

```
                    inc(CHECKER);
                end;
                if CHECKER > 0 then J := SIZEAT;
                inc(J);
            end;
            if CHECKER > 0 then I := SIZEBT;
            inc(I);
        end;
 end;

{Fill in the THI matrix for the Y's specified in previous modelling}
 CHECKER := 1;
 while CHECKER <> 0 do begin
        CHECKER := 0;
        MARK := SIZEAT*SIZEBT;
        I := 1;
        while I <= SIZEBT  do begin
            J := 1;
            while J <= SIZEAT do begin
                inc(MARK);
                if MTEMP[SIZEAT+I,J] = 1 then begin
                   MTEMP[SIZEAT+I,J] := 3;
                   thiyes(MREACHT,TOBEDEL,THI,MARK,N2SIZE,SIZEAT,I);
                   inc(CHECKER);
                end else if MTEMP[SIZEAT+I,J] = 0 then begin
                   MTEMP[SIZEAT+I,J] := 3;
                   thino(MREACHT,TOBEDEL,THI,MARK,N2SIZE,SIZEAT,I);
                   inc(CHECKER);
                end;
                if CHECKER > 0 then J := SIZEAT;
                inc(J);
            end;
            if CHECKER > 0 then I := SIZEBT;
            inc(I);
        end;
 end;
{set start point at end of THI matrix and count backward first through
the Y then through the X matrix, deleting the modelled elements.}
 RCR := 2*SIZEAT*SIZEBT;
 For I := SIZEBT downto 1 do begin
     for J := SIZEAT downto 1 do begin
         if MREACHT[SIZEAT+I,J] < 2 then begin
            dimin(THI,N2SIZE,RCR);
         end;
         dec(RCR);
     end;
 end;
 for I := SIZEBT downto 1 do begin
     for J := SIZEAT downto 1 do begin
         if MREACHT[J,SIZEAT+I] < 2 then begin
            dimin(THI,N2SIZE,RCR);
         end;
         dec(RCR);
     end;
 end;
end;
```

```
Procedure thigeneral(B : tarr5 ; SIZEBT : integer);
{Procedure to find the THI matrix and to solve for it for the general
case, where the values of the X set above the B matrix are unknown.}
var
   BBAR,BTRANS : tarr5;

begin
{find the matrix N1}
 complementB(B,BBAR,SIZEBT);
 transposeB(B,BTRANS,SIZEBT);
 booladdb(BBAR,SIZEBT);
 for I := 1 to 2*SIZEBT do begin
     for J:=1 to 2*SIZEBT do begin
         n1[I,J] := 2;
     end;
 end;
 for I := 1 to SIZEBT do begin
     for J := 1 to SIZEBT do begin
         N1[I,J] := BTRANS[I,J];
         N1[SIZEBT+I,J] := BBAR[I,J];
         N1[I,SIZEBT+J] := 0;
         N1[SIZEBT+I,SIZEBT+J] := BTRANS[I,J];
     end;
 end;
 N1SIZE := 2*SIZEBT;
end;
```

```
Procedure findN2(var THI : tarr100; var N2SIZE : integer; N1 : tarr5;
                 A : tarr51; N1SIZE : integer; SIZEAT : shortint);
var
   ATRANS,ABAR,II : tarr51;

begin
{find the matrix N2, from the matrix N1, and from the A matrix.}
 transpA(A,ATRANS,SIZEAT);
 complA(ABAR,ATRANS,SIZEAT);
 makei(II);
 M:=0;
 N2SIZE := n1size*SIZEAT;
 for I:= 1 to N2SIZE do begin
     for J:= 1 to N2SIZE do begin
         THI[I,J] := 0;
     end;
 end;
 for I := 1 to N1SIZE do begin
      for J := 1 to N1SIZE do begin
          If N1[I,J] = 0 then begin
             for K := 1 to SIZEAT do begin
                 for L := 1 to SIZEAT do begin
                     THI[SIZEAT*(I-1)+K,SIZEAT*(J-1)+L] := 0;
                 end;
             end;
             M := 1;
          end;
          if M = 0 then begin
             if I = J then begin
                for K := 1 to SIZEAT do begin
                    for L := 1 to SIZEAT do begin
                        THI[(SIZEAT*(I-1)+K),(SIZEAT*(J-1)+L)] := A[K,L];
                    end;
                end;
                M := 1;
             end;
          end;
          if M = 0 then begin
             if J = I-SIZEBT then begin
                 for K := 1 to SIZEAT do begin
                     for L := 1 to SIZEAT do begin
                         THI[SIZEAT*(I-1)+K,SIZEAT*(J-1)+L] := ABAR[K,L];
                     end;
                 end;
                 M := 1;
             end;
          end;
          if M = 0 then begin
              for K := 1 to SIZEAT do begin
                 for L := 1 to SIZEAT do begin
                     THI[SIZEAT*(I-1)+K,SIZEAT*(J-1)+L] := II[K,L];
                 end;
              end;
          end;
          M := 0
      end;
 end;
end;
```

```
Procedure thifind(var MREACH : tarr51; var MRSIZE : shortint;
          SIZEA,SIZEB,special : shortint; CONTEXT,RELATE : string;
          ELEMENT : tars51);
var
   THI : tarr100;

begin
 MRSIZET := MRSIZE;
 SIZEAT := SIZEA;
 SIZEBT := SIZEB;
{limit the size of matrix B due to TP's memory restrictions to the size of
THI}
 I:=100 div (2*SIZEAT);
 if I < SIZEBT then SIZEBT :=I;
 if SIZEBT > 15 then SIZEBT := 15;
 for I := 0 to MRSIZET do begin
     move(MREACH[I],MREACHT[I],MRSIZET+1);
 end;
 for I := 1 to SIZEAT do begin
     for J:= 1 to SIZEAT do begin
         A[I,J] := MREACHT[I,J];
     end;
 end;
 for I := SIZEA+1 to SIZEA+SIZEBT do begin
     for J:= SIZEA + 1 to SIZEA+SIZEBT do begin
         B[I-SIZEA,J-SIZEA] := MREACHT[I,J];
     end;
 end;
 thigeneral(b,SIZEBT);
 findn2(THI,N2SIZE,N1,A,N1SIZE,SIZEAT);
 findthi(THI,N2SIZE,SIZEAT,SIZEBT);
 fillthi(THI,N2SIZE,ELEMENT,CONTEXT,RELATE);
 MRSIZE := MRSIZET;
 move(MREACHT[0],MREACH[0],MRSIZE+1);
 for I := 1 to MRSIZE do begin
     move(MREACHT[I],MREACH[I],MRSIZE+1);
 end;
end;
end.
```

PASSUNIT

```
Unit passunit;
{unit to structure the matrix for further modelling on a new matrix where
{no prior modelling has occurred}

Interface

uses
      crt,typeunit;

Procedure passone(var MREACH : tarr51; var FEEDARR : tar51;
                  var FEEDCNTR,POSITION,MRSIZE : shortint;
                  ELEMENT : tars51; CONTEXT,RELATE : string);

Implementation

var
   PASS,TEMPCNT,REMCNT          : shortint;
   Q,CHECKER,LIFTCNT            : shortint;
   FEEDCNT,DROPCNT,VACCNT       : shortint;
   I,J,K,L,M,COUNTER,SIZE       : shortint;
   BOOL                         : char;

Procedure plift(var MREACH : tarr51; var TEMP,REM : tar51;
                var LIFTCNT,FEEDCNT,DROPCNT,VACCNT,TEMPCNT,REMCNT,
                POSITION : shortint; ELEMENT : tars51;
                CONTEXT,RELATE : string);
begin
{find LIFT set}
 LIFTCNT := 0;
 FEEDCNT := 0;
 DROPCNT := 0;
 VACCNT := 0;
 TEMPCNT := 0;
 REMCNT := 0;
 for I := 2 to SIZE do begin
     clrscr;
     gotoxy(1,5);
     writeln(CONTEXT);
     writeln;
     writeln(ELEMENT[MREACH[POSITION+1,0]]);
     writeln;
     writeln(RELATE);
     writeln;
     writeln(ELEMENT[MREACH[POSITION+I,0]]);
     writeln;
     write('Y/N? ');
     BOOL :=readyn;
     dellay;
     clrscr;
     case BOOL of
          'Y' : begin
                inc(TEMPCNT);
                MREACH[POSITION+1,POSITION+I] := 1;
                TEMP[TEMPCNT] := POSITION+I;
          end;
          'N' : begin
                MREACH[POSITION+1,POSITION+I] := 0;
                inc(REMCNT);
                REM[REMCNT] := POSITION+I;
          end;
     end;
 end;
end;
```

```
Procedure pfeed(var MREACH : tarr51; var FEED,LIFT,TEMP : tar51;
                var FEEDCNT,LIFTCNT,TEMPCNT,POSITION : shortint;
                ELEMENT : tars51; CONTEXT,RELATE : string);
{find feedback set}
begin
 FEEDCNT := 0;
 LIFTCNT := 0;
 For I := 1 to TEMPCNT do begin
     clrscr;
     gotoxy(1,5);
     writeln (CONTEXT);
     writeln;
     writeln (ELEMENT[MREACH[TEMP[I],0]]);
     writeln;
     writeln (RELATE);
     writeln;
     writeln (ELEMENT[MREACH[POSITION+1,0]]);
     writeln;
     write('Y/N? ');
     BOOL :=readyn;
     dellay;
     clrscr;
     case BOOL of
          'Y' : begin
                inc(FEEDCNT);
                MREACH[TEMP[I],POSITION+1] := 1;
                FEED[FEEDCNT] := MREACH[TEMP[I],0];
          end;
          'N' : begin
               inc(LIFTCNT);
               LIFT[LIFTCNT] := TEMP[I];
               MREACH[TEMP[I],POSITION+1] := 0;
          end;
     end;
 end;
end;
```

```
Procedure pdrop(var MREACH : tarr51; var DROP,VAC,REM : tar51;
                var DROPCNT,VACCNT,REMCNT,POSITION : shortint;
                ELEMENT : tars51; CONTEXT,RELATE : string);
{find drop and vacancy sets}
begin
 DROPCNT := 0;
 VACCNT := 0;
 For I := 1 to REMCNT do begin
     clrscr;
     gotoxy(1,5);
     writeln (CONTEXT);
     writeln;
     writeln (ELEMENT[MREACH[REM[I],0]]);
     writeln;
     writeln (RELATE);
     writeln;
     writeln (ELEMENT[MREACH[POSITION+1,0]]);
     writeln;
     write('Y/N? ');
     BOOL :=readyn;
     dellay;
     clrscr;
     case BOOL of
          'Y' : begin
                inc(DROPCNT);
                MREACH[REM[I],POSITION+1] := 1;
                DROP[DROPCNT] := REM[I];
          end;
          'N' : begin
                inc(VACCNT);
                MREACH[REM[I],POSITION+1] := 0;
                VAC[VACCNT] := REM[I];
          end;
     end;
 end;
end;
```

```
procedure rowscolumns(var MREACH : tarr51; var LIFT,VAC,DROP : tar51;
                      var LIFTCNT,VACCNT,DROPCNT,POSITION,MRSIZE : shortint);
var
   MTEMP : tarr51;
begin
{copy rows into temporary matrix, and then sort the columns as the
temporary matrix is copied back to the original matrix.}
 for I := 0 to POSITION do
     move(MREACH[I],MTEMP[I],MRSIZE+1);
 for I := 1 to LIFTCNT do
     move(MREACH[LIFT[I]],MTEMP[POSITION+I],MRSIZE+1);
     move(MREACH[POSITION+1],MTEMP[POSITION+LIFTCNT+1],MRSIZE+1);
 for I := 1 to VACCNT do
     move(MREACH[VAC[I]],MTEMP[POSITION+LIFTCNT+I+1],MRSIZE+1);
 for I := 1 to DROPCNT do
     move(MREACH[DROP[I]],MTEMP[POSITION+LIFTCNT+VACCNT+1+I],MRSIZE+1);
 if (POSITION+LIFTCNT+DROPCNT+VACCNT+FEEDCNT+1) < MRSIZE then begin
    for I := (POSITION+LIFTCNT+DROPCNT+VACCNT+2) to (MRSIZE+FEEDCNT) do
        move(MREACH[I+FEEDCNT],MTEMP[I],MRSIZE+1);
 end;
{copy rows from temporary matrix back}
 for I := 0 to MRSIZE do begin
     for J := 0 to MRSIZE do begin
         MREACH[i,j] :=2;
     end;
 end;
 for J := 0 to POSITION do begin
       for I := 0 to MRSIZE do begin
           MREACH[I,J] := MTEMP[I,J];
       end;
 end;
 for J := 1 to LIFTCNT do begin
       for I := 0 to MRSIZE do begin
           MREACH[I,POSITION+J] := MTEMP[I,LIFT[J]];
       end;
 end;
 for I := 0 to MRSIZE do begin
     MREACH[I,POSITION+LIFTCNT+1] := MTEMP[I,POSITION+1];
 end;
 for J := 1 to VACCNT do begin
       for I := 0 to MRSIZE do begin
           MREACH[I,POSITION+1+LIFTCNT+J] := MTEMP[I,VAC[J]];
       end;
 end;
 for J := 1 to DROPCNT do begin
       for I := 0 to MRSIZE do begin
           MREACH[I,POSITION+LIFTCNT+VACCNT+1+J] := MTEMP[I,DROP[J]];
       end;
 end;
 if (POSITION+LIFTCNT+DROPCNT+FEEDCNT+VACCNT+1) < MRSIZE then begin
    for J := (POSITION+LIFTCNT+DROPCNT+VACCNT+2) to (MRSIZE-FEEDCNT) do begin
        for I := 0 to MRSIZE do begin
            MREACH[I,J] := MTEMP[I,J+FEEDCNT];
        end;
    end;
 end;
end;
```

```
procedure inferred(var MREACH : tarr51; var LIFTCNT,VACCNT,DROPCNT,
                   POSITION,MRSIZE : shortint);
{fill in the inferred 1's and 0's}
begin
 For I := 1 to MRSIZE do
     MREACH[I,I] := 1;
 For I := 1 to LIFTCNT do begin
       for J := 1 to (VACCNT+DROPCNT) do begin
           MREACH[POSITION+I,LIFTCNT+1+POSITION+J] := 0;
       end;
 end;
 For I := 1 to VACCNT do begin
     for J := 1 to DROPCNT do begin
         MREACH[POSITION+LIFTCNT+1+I,POSITION+LIFTCNT+VACCNT+1+J] := 0;
     end;
 end;
 For I := 1 to DROPCNT do begin
     for J := 1 to LIFTCNT do begin
         MREACH[POSITION+LIFTCNT+VACCNT+1+I,POSITION+J] := 1;
     end;
 end;
end;

procedure filarr(var LIFTARR,PARTARR,DROPARR,VACARR : tarr1;
                 var pass,POSITION,LIFTCNT,DROPCNT,VACCNT : shortint);
{procedure to store lift, drop and vac sizes an start points and lengths}
begin
 LIFTARR[PASS,1] := POSITION + 1;
 LIFTARR[PASS,2] := LIFTCNT;
 PARTARR[PASS,1] := POSITION + LIFTCNT+ 1;
 PARTARR[PASS,2] := 1;
 DROPARR[PASS,1] := (POSITION + VACCNT + LIFTCNT + 2);
 DROPARR[PASS,2] := DROPCNT;
 VACARR[PASS,1]  := (POSITION + LIFTCNT + 2);
 VACARR[PASS,2]  := VACCNT;
end;
Procedure ppasi(var MREACH : tarr51; var LIFTARR,PARTARR,VACARR,
              DROPARR : tarr1; var LIFT,FEED,VAC,DROP,FEEDARR : tar51;
                var LIFTCNT,VACCNT,DROPCNT,FEEDCNT,MRSIZE,POSITION,
              FEEDCNTR : shortint; ELEMENT: tars51;
              CONTEXT,RELATE : string);
{Control procedure for the Passone/ Scanning method of starting modelling.}
var
   TEMP,REM : tar51;
begin
 plift(MREACH,TEMP,REM,LIFTCNT,FEEDCNT,DROPCNT,VACCNT,TEMPCNT,REMCNT,
      POSITION,ELEMENT,CONTEXT,RELATE);
 pfeed(MREACH,FEED,LIFT,TEMP,FEEDCNT,LIFTCNT,TEMPCNT,POSITION,ELEMENT,
      CONTEXT,RELATE);
 pdrop(MREACH,DROP,VAC,REM,DROPCNT,VACCNT,REMCNT,POSITION,ELEMENT,
      CONTEXT,RELATE);
{capture the feedback set}
 if FEEDCNT > 0 then begin
    inc(FEEDCNTR);
    FEEDARR[FEEDCNTR] := FEEDCNT+1;
    FEEDARR[FEEDCNTR+1] := MREACH[POSITION+1,0];
    for I := 2 to (FEEDCNT+1) do begin
        FEEDARR[FEEDCNTR+I] := FEED[I-1];
    end;
    FEEDCNTR := FEEDCNTR + FEEDCNT + 1;
 end;
 rowscolumns(MREACH,LIFT,VAC,DROP,LIFTCNT,VACCNT,DROPCNT,POSITION,MRSIZE);
 inferred(MREACH,LIFTCNT,VACCNT,DROPCNT,POSITION,MRSIZE);
end;
```

PASSUNIT

```
Procedure passone(var MREACH : tarr51; var FEEDARR : tar51;
          var FEEDCNTR,POSITION,MRSIZE : shortint; ELEMENT : tars51;
          CONTEXT,RELATE : string);
var
   LIFTARR,PARTARR,DROPARR,VACARR : tarr1;
   LIFT,FEED,VAC,DROP             : tar51;
{Main Program to control pass loops in passone and sortone}
begin
{Initialize - copy to unit variable names, global within unit}
 SIZE := MRSIZE-POSITION;
 LIFTCNT :=0;
 FEEDCNT:=0;
 DROPCNT:=0;
 VACCNT:=0;
 PASS := 0;
 Q:=0;
 ppasi(MREACH,LIFTARR,PARTARR,VACARR,DROPARR,LIFT,FEED,VAC,DROP,FEEDARR,
      LIFTCNT,VACCNT,DROPCNT,FEEDCNT,MRSIZE,POSITION,FEEDCNTR,ELEMENT,
      CONTEXT,RELATE);
 filarr(LIFTARR,PARTARR,DROPARR,VACARR,PASS,POSITION,LIFTCNT,DROPCNT,VACCNT);
 while Q<>1 do begin
  CHECKER := 0;
  {decrease matrix size by the size of the feedback set.}
  if FEEDCNT > 0 then MRSIZE := MRSIZE - FEEDCNT;
{deal with the lift matrix}
    if LIFTARR[PASS,1] = (POSITION + 1) then begin
       if LIFTARR[PASS,2] > 1 then begin
          SIZE := LIFTARR[PASS,2];
          inc(PASS);
          ppasi(MREACH,LIFTARR,PARTARR,VACARR,DROPARR,LIFT,FEED,VAC,DROP,
              FEEDARR,LIFTCNT,VACCNT,DROPCNT,FEEDCNT,MRSIZE,POSITION,
              FEEDCNTR,ELEMENT,CONTEXT,RELATE);
          CHECKER := 1;
{for the feedback sets, account must be taken of previously stored variables
in vacarr etc. The start of these must be decreased so as to match up later
when comparing to position.}
          if FEEDCNT > 0 then begin
             for I := (PASS-1) downto 0 do begin
                 PARTARR[I,1] := PARTARR[I,1] - FEEDCNT;
                 VACARR[I,1] := VACARR[I,1] - FEEDCNT;
                 DROPARR[I,1] := DROPARR[I,1] - FEEDCNT;
             end;
          end;
       end else begin
           POSITION := POSITION + LIFTARR[PASS,2];
       end;
    end;
    if PARTARR[PASS,1] = (POSITION + 1) then inc(POSITION);
{deal with the vacancy matrix}
    if CHECKER = 0 then begin
       if VACARR[PASS,1] = (POSITION + 1) then begin
          if VACARR[PASS,2] > 1 then begin
             SIZE := VACARR[PASS,2];
             inc(PASS);
            ppasi(MREACH,LIFTARR,PARTARR,VACARR,DROPARR,LIFT,FEED,VAC,DROP,
                FEEDARR,LIFTCNT,VACCNT,DROPCNT,FEEDCNT,MRSIZE,POSITION,
                FEEDCNTR,ELEMENT,CONTEXT,RELATE);
             CHECKER := 1;
{for the feedback sets, account must be taken of previously stored variables
in vacarr etc. The start of these must be decreased so as to match up later
when comparing to position.}
             if FEEDCNT > 0 then begin
                DROPARR[PASS-1,1] := DROPARR[PASS-1,1] - FEEDCNT;
                for I := (PASS-2) downto 0 do begin
```

```
                    PARTARR[I,1] := PARTARR[I,1] - FEEDCNT;
                    VACARR[I,1] := VACARR[I,1] - FEEDCNT;
                    DROPARR[I,1] := DROPARR[I,1] - FEEDCNT;
                end;
             end;
          end else begin
              POSITION := POSITION + VACARR[PASS,2];
          end;
       end;
    end;
{deal with the drop matrix}
    if CHECKER = 0 then begin
       if DROPARR[PASS,1] = (POSITION + 1) then begin
          if DROPARR[PASS,2] > 1 then begin
             SIZE := DROPARR[PASS,2];
             inc(PASS);
             ppasi(MREACH,LIFTARR,PARTARR,VACARR,DROPARR,LIFT,FEED,VAC,
                 DROP,FEEDARR,LIFTCNT,VACCNT,DROPCNT,FEEDCNT,MRSIZE,
                 POSITION,FEEDCNTR,ELEMENT,CONTEXT,RELATE);
             CHECKER := 1;
{for the feedback sets, account must be taken of previously stored variables
in vacarr etc. The start of these must be decreased so as to match up later
when comparing to position.}
             if FEEDCNT > 0 then begin
                for I := (PASS-2) downto 0 do begin
                    PARTARR[I,1] := PARTARR[I,1] - FEEDCNT;
                    VACARR[I,1] := VACARR[I,1] - FEEDCNT;
                    DROPARR[I,1] := DROPARR[I,1] - FEEDCNT;
                end;
             end;
          end else begin
              POSITION := POSITION + DROPARR[PASS,2];
          end;
       end;
    end;
{deal with needing to drop one level}
    if CHECKER = 0 then begin
       if POSITION >= MRSIZE-1 then begin
          Q := 1;
          exit;
       end;
       LIFTCNT :=0;
       FEEDCNT:=0;
       DROPCNT:=0;
       VACCNT:=0;
       dec(PASS);
    end else begin
        filarr(LIFTARR,PARTARR,DROPARR,VACARR,PASS,POSITION,LIFTCNT,DROPCNT,
             VACCNT);
    end;
  end;
 end;
end.
```

DIGUNIT

```
Unit DIGUNIT;
{Unit to find the skeleton matrix of mreach and to plot the digraph.}

Interface
 uses
     crt,typeunit,graph,bgidriv,bgifont;

Procedure digraph(MREACH : tarr51; ELEMENT : tars51; FEEDARR : tar51;
                  FEEDCNTR,MRSIZE : shortint);

Implementation

type
    tarrr651 = array[0..6,0..50,0..50] of shortint;
    tarr351  = array[1..50,1..3] of shortint;
    tarri351 = array[0..50,1..3] of integer;

var
   MDIG,MSKEL                          : tarr51;
   Z,X1,I,J,K,L,M,N,R,CHECKER          : shortint;
   SIZE,PROBLEM,CHOICE                 : shortint;
   COMPLETE,POWER,DIAGONAL             : shortint;
   TEXTSIZE,Y1                         : real;
   BOOL                                : char;
   COUNT,CSIZE,MAXX,MAXY               : integer;
   GRAPHDRIVER,GRAPHMODE,ERRORCODE     : integer;
   RAD,SPACE,P,Q,X,Y,RAD1,RAD2         : integer;
   FROMX,FROMY,TOX,TOY                 : integer;
   S,TEMPS                             : string;

procedure multiply(var MTEMP : tarr51; MPOWER : tarrr651; M,SIZE:shortint);
{Procedure to Multiply two matrices.}
var
   MJUNK : tarr51;

begin
     For I := 1 to SIZE do begin
         for J := 1 to SIZE do begin
             MJUNK[I,J] := 0;
         end;
     end;
     for I := 1 to SIZE do begin
         for J := 1 to SIZE do begin
             for K := 1 to SIZE do begin
                 if MJUNK[I,J] = 1 then break;
                 MJUNK[I,J] := MJUNK[I,J] + MTEMP[I,K]*MPOWER[M,K,J];
             end;
         end;
     end;
     for I := 0 to SIZE do begin
         move(MJUNK[I],MTEMP[I],SIZE+1);
     end;
end;

Procedure same2(var MPOWER : tarrr651;MTEMP : tarr51; R,SIZE:shortint);
{Procedure to make two matrices equal.}
begin
     for I := 0 to SIZE do begin
         move(MPOWER[R,I],MTEMP[I],SIZE+1);
     end;
end;
```

```
Procedure same3(var MTEMP : tarr51; var MPOWER : tarrr651; R,SIZE:shortint);
{Procedure to make two matrices equal.}
begin
     for I := 0 to SIZE do begin
         move(MTEMP[I],MPOWER[R,I],SIZE+1);
     end;
end;

Procedure MTON(var MTEMP : tarr51; SIZE : shortint);
{procedure to raise matrix m to the power n}
var
   MPOWER : TARRR651;
begin
X1:=0;
  POWER := (SIZE-1);
  Y1 := power;
  X := 1;
  while Y1 >= 2 do begin
        inc(X1);
        X := X*2;
        Y1 := Y1/2;
  end;
  R := 0;
  same3(MTEMP,MPOWER,R,SIZE);
  for R := 1 to (X1) do begin
       M := R - 1;
      multiply(MTEMP,MPOWER,M,SIZE);
      thinking;
      same3(MTEMP,MPOWER,R,SIZE);
  end;
  R := X1;
  POWER := POWER - X;
  While POWER <> 0 do begin
        X1 := 0;
        Y1 := POWER;
        X := 1;
        while Y1 >= 2 do begin
           inc(X1);
           X := X*2;
           Y1 := Y1/2;
        end;
  M := X1;
  multiply(MTEMP,MPOWER,M,SIZE);
  POWER := POWER - X;
  Y1 := POWER;
  If Y1 = 1 then begin
     M := 0;
     multiply(MTEMP,MPOWER,M,SIZE);
     POWER := 0;
  end;
  end;
end;

Procedure same(var MTEMP,MSKEL : tarr51; SIZE:integer);
{Procedure to make two matrices equal.}
begin
     for I := 0 to SIZE do begin
         move(MSKEL[I],MTEMP[I],SIZE+1);
     end;
end;
```

```
Procedure check(MDIG : tarr51;SIZE : integer);
{Check if modelling complete, and change 2's to 0's if not}
begin
 I := 1;
 while I <= SIZE do begin
     for J := 1 to SIZE do begin
         if MDIG[I,J] = 2 then begin
            writeln;
            Writeln('The modelling of the elements is incomplete.');
            writeln;
            writeln('The digraph drawn will show only the existing links.');
            writeln;
            Write('                          Press any key to continue.');
            Readkey;
            clrscr;
            for K := 1 to SIZE do begin
                for L:= 1 to SIZE do begin
                    if MDIG[K,L] = 2 then MDIG[I,J] :=0;
                    I := SIZE + 1;
                end;
            end;
          end;
      end;
    inc(I);
 end;
end;
```

```
Procedure main(var MDIG,MSKEL : tarr51; SIZE,PROBLEM : shortint);
{Main program to find skeleton matrix of mreach.}
var
   MERR,MTEMP   : tarr51;
begin
 For I := 1 to SIZE do begin
     for J := 1 to SIZE do begin
         MSKEL[I,J] := 0;
     end;
 end;
{Check(MDIG,SIZE);}
 for I := 1 to SIZE do begin
     MSKEL[I,I] := MDIG[I,I];
     MSKEL[I,0] := MDIG[I,0];
     MSKEL[0,I] := MDIG[0,I];
 end;
 for I := 1 to SIZE-1 do begin
     MSKEL[I+1,I] := MDIG[I+1,I];
 end;
 DIAGONAL := 1;
 PROBLEM := 2;
 if size = 1 then PROBLEM := 1;
 While PROBLEM <> 1 do begin
       thinking;
       PROBLEM := 1;
       same(MTEMP,MSKEL,SIZE);
       mton(MTEMP,SIZE);
       I := 1;
       J := 1;
       while I <= SIZE do begin
             while J <= SIZE do begin
                   if MDIG[J,I] - MTEMP[J,I] = 1 then begin
                      inc(PROBLEM);
                      for P := 1 to SIZE do begin
                          for Q := 1 to SIZE do begin
                              MERR[P,Q] := MDIG[P,Q] - MTEMP[P,Q];
                          end;
                      end;
                      inc(DIAGONAL);
                      for K := 1 to (SIZE-DIAGONAL) do begin
                          MSKEL[K+DIAGONAL,K] := MERR[K+DIAGONAL,K];
                      end;
                      I := SIZE;
                      J := SIZE;
                   end;
                   inc(J);
             end;
             J := I+1;
             inc(I);
       end;
 end;
end;

Procedure dimin2(var MTEMP:tarr51; var BSIZE : shortint; RCD : shortint);
{proceedure to diminish the size of mtemp}
begin
     for X := (RCD+1) to BSIZE do begin
         move(MTEMP[X],MTEMP[X-1],BSIZE+1);
         for Y := 0 to (BSIZE+1) do begin
             MTEMP[Y,X-1] := MTEMP[Y,X];
         end;
     end;
      dec(BSIZE);
end;
```

```
Procedure blocklevel(var MDIG : tarr51; var LCOUNT : tar51; CHOICE :
shortint);
{Procedure to blocklevel the mreach matrix}
var
   LEVEL : tarr351;
   MTEMP : tarr51;
   RCD,N,BSIZE : shortint;
   LTEMP : array[0..50] of shortint;

begin
 Check(MDIG,SIZE);
 For I := 0 to SIZE do begin
     move(MDIG[I],MTEMP[I],SIZE+1);
     LCOUNT[I] := 0;
 end;
 for I := 1 to SIZE do begin
     MTEMP[I,0] := I;
     MTEMP[0,I] := I;
 end;
 K := 1;
 L := 0;
 M := 0;
 N := 0;
 BSIZE := SIZE;
 if CHOICE = 1 then begin
  While BSIZE > 0 do begin
       for I := 1 to BSIZE do begin
           for J := 1 to BSIZE do begin
               If MTEMP[I,J] = 1 then if MTEMP[J,I] = 0 then inc(L)
           end;
           If L = 0 then begin
              inc(M);
              LEVEL[M,1] := K;
              LEVEL[M,2] := mtemp[0,I];
              LEVEL[M,3] := I;
           end;
           L := 0;
       end;
       For I := M downto (N+1) do begin
           RCD := LEVEL[I,3];
           dimin2(MTEMP,BSIZE,RCD);
           inc(LCOUNT[K]);
       end;
  inc(K);
  N := M;
  LCOUNT[0] := K-1;
  end;
 end else begin
  While BSIZE > 0 do begin
       for I := 1 to BSIZE do begin
           for J := 1 to BSIZE do begin
               If MTEMP[J,I] = 1 then if MTEMP[I,J] = 0 then inc(L)
           end;
           If L = 0 then begin
              inc(M);
              LEVEL[M,1] := K;
              LEVEL[M,2] := MTEMP[0,I];
              LEVEL[M,3] := I;
           end;
       L := 0;
       end;
       For I := M downto (N+1) do begin
           RCD := LEVEL[I,3];
           dimin2(MTEMP,BSIZE,RCD);
```

```
            inc(LCOUNT[K]);
          end;
      inc(K);
      N := M;
      LCOUNT[0] := K-1;
      end;
    end;
    Move(MDIG[0],MTEMP[0],SIZE+1);
    for I := 1 to SIZE do begin
        move(MDIG[LEVEL[I,2]],MTEMP[I],SIZE+1);
    end;
    for I := 1 to SIZE do begin
        MDIG[I,0] := MTEMP[I,0];
    end;
    for I := 0 to SIZE do begin
        For J := 1 to SIZE do begin
         MDIG[I,J] := MTEMP[I,LEVEL[J,2]];
        end;
    end;
    if CHOICE = 2 then begin
       for I := 0 to 50 do begin
           LTEMP[I] := LCOUNT[I];
       end;
       for I := 1 to LCOUNT[0] do begin
           LCOUNT[I] := LTEMP[(LCOUNT[0]-I+1)];
       end;
       for I := 0 to SIZE do begin
           move(MDIG[I],MTEMP[I],SIZE+1);
       end;
       for I := 1 to SIZE do begin
           MDIG[I,0] := MTEMP[SIZE-I+1,0];
           MDIG[0,I] := MTEMP[0,SIZE-I+1];
       end;
       for I := 1 to SIZE do begin
           for J := 1 to SIZE do begin
               MDIG[I,J] := MTEMP[SIZE-I+1,SIZE-J+1];
           end;
       end;
    end;
  end;
```

```
Procedure feedin(var MDIG,MSKEL : tarr51; var LCOUNT,FEEDARR : tar51;
                 var SIZE,FEEDCNTR : shortint);
{Procedure to expand the matrix to include feedback sets.}
var
   MTEMP : TARR51;

begin
    CHECKER := 1;
    M := 2;
    N := 1;
    While CHECKER < FEEDCNTR do begin
       L := 0;
       I := 1;
       While I <= SIZE do begin
             If M < FEEDCNTR then begin
             If MSKEL[0,I+L] = FEEDARR[M] then begin
                inc(M);
                For Q := 1 to (FEEDARR[N]-1) do begin
                    for J := (SIZE+L+Q) downto I+L+Q do begin
                        move(MSKEL[J-1],MSKEL[J],SIZE+L+FEEDARR[N]-1);
                    end;
                    for J := (SIZE+L+Q) downto I+L+Q do begin
                        for P:= 0 to (SIZE+L+Q) do begin
                            MSKEL[P,J] := MSKEL[P,J-1];
                        end;
                    end;
                    P := I+L;
                    for R := 1 to LCOUNT[0] do begin
                        P := P - LCOUNT[R];
                        if P <= 0 then begin
                           inc(LCOUNT[R]);
                           P := P + 100;
                        end;
                    end;
                    MSKEL[0,I+Q+L] := FEEDARR[M];
                    MSKEL[I+Q+L,0] := FEEDARR[M];
                    inc(M);
                end;
             L := L + (FEEDARR[N]-1);
             CHECKER := CHECKER + (FEEDARR[N]+1);
             N := M;
             inc(M);
             end;
             end;
       inc(I);
       end;
 SIZE := SIZE + L;
 end;
end;
```

```
Procedure circles(var POINTERS : tarri351; MSKEL : tarr51; LCOUNT : tar51;
                  RAD,SPACE : SHORTINT);
begin
 K := 0;
 MAXX := getmaxx;
 MAXY := getmaxy-32;
 X := 0;
 For I := LCOUNT[0] downto 1 do begin
     Y := MAXY div 2 - round((LCOUNT[I] div 2 + 1)*1.5*SPACE);
     If LCOUNT[I] mod 2 = 0 then Y := round(Y + 0.75*SPACE);
     X := X + 3*SPACE;
     for J := 1 to LCOUNT[I] do begin
         Y := Y + round(1.5*SPACE);
         circle(X,Y,RAD);
         settextstyle(2,HORIZDIR,CSIZE);
         L := MSKEL[SIZE-K,0];
         str(L,S);
         L := textheight(S);
         rad1:=11*L div 16;
         L := textwidth(S);
         rad2 := 7*L div 16;
         outtextxy(X-RAD2,Y-RAD1,s);
{store values of x&y for later plotting}
         POINTERS[K,1] := MSKEL[SIZE-K,0];
         POINTERS[K,2] := X;
         POINTERS[K,3] := Y;
         inc(K);
     end;
 end;
end;

Procedure linedraw(MSKEL : tarr51; POINTERS : tarri351; SIZE : shortint;
                   RAD,SPACE : SHORTINT);
begin
 For I := 2 to SIZE do begin
     for J := 1 to I-1 do begin
         If MSKEL[I,J] = 1 then
         begin
            FROMX := POINTERS[SIZE-I,2]+RAD;
            FROMY := POINTERS[SIZE-I,3];
            TOX := POINTERS[SIZE-J,2]-RAD;
            TOY := POINTERS[SIZE-J,3];
            if (FROMX-TOX)/2 = RAD then
            begin
                 TOY := TOY - RAD;
                 TOX := TOX + RAD;
                 FROMX := FROMX - RAD;
                 FROMY := FROMY + RAD;
            end;
            line(FROMX,FROMY,TOX,TOY);
         end;
     end;
 end;
end;
```

```
Procedure outline(FEEDARR : tar51; FEEDCNTR : shortint; POINTERS : tarri351;
                  RAD,SPACE : SHORTINT);
begin
 I := 1;
 While I < FEEDCNTR do begin
       J := 1;
       while J <= SIZE do begin
           if FEEDARR[I+1] = 0 then break;
           if FEEDARR[I+1] = POINTERS[J,1] then
           begin
                setlinestyle(3,$AAAA,normwidth);
                X := round(POINTERS[J,2]+round(1.2*rad));
                Y := round(POINTERS[J,3]+0.75*SPACE);
                Q := round(Y-FEEDARR[I]*1.5*SPACE);
                P := round(X-round(2.4*rad));
                moveto(X,Y);
                rectangle(X,Y,P,Q);
                I := I + FEEDARR[I]+1;
                setlinestyle(0,$AAAA,normwidth);
                if I >= FEEDCNTR then J := SIZE;
           end;
           inc(J);
       end;
 end;
end;

function valid( key: char ): boolean;
  begin
    VALID := false;
    if COUNT = 0 then begin
       if( ((KEY >= '0') and ( KEY <= '9')) or
           (ord(KEY) = 8) or       {delete <-}
           (ord(KEY) = 13) or      {return}
           ((KEY >= 'O') and (KEY <= 'R')) or
           ((KEY >= 'o') and (KEY <= 'r')) or
           (KEY = 'E') or (KEY = 'e') or
           (KEY = 'C') or (KEY = 'c'))
            then
           VALID := true
       else
           write( chr( 7 ) );
    end;
    if count <> 0 then begin
       if( ((KEY >= '0') and ( KEY <= '9')) or
           (ord(KEY) = 8) or
           (ord(KEY) = 13) )
           then
           VALID := true
       else
           write( chr( 7 ) );
    end;
  end;

function getkey: char;
var
    CH : char;

begin
    CH := readkey;
    while (not(valid(CH))) do
            CH := readkey;
            getkey := CH;
      end;
end;
```

```
function get_choice: string;
var
    SELECT,S          : string;
    XX,YY,I,CODE      : integer;
    KEY               : char;

begin
    COMPLETE := 0;
    SELECT := '';
    COUNT := 0;
    XX := 0;
    YY := MAXY-3*L;        { Get start position }
    while COMPLETE <> 1 do
    begin
    KEY := getkey;
      case ord( KEY ) of
        8  : begin                  { This sorts out the backspace }
               if( COUNT <= 0 ) then
                 write( chr( 7 ) )
               else
                 begin
                   XX := XX - 8;
                   moveto(XX,YY);
                   setcolor(BLACK);
                   outtextxy(XX,YY,chr(219));
                   setcolor(WHITE);
                   delete(SELECT,COUNT-1,255);
                   COUNT := COUNT - 1;
                 end;
             end;
        13 : COMPLETE := 1;
      else
        begin
          SELECT := concat(SELECT,KEY);
          outtextxy(XX,YY,KEY);
          COUNT := COUNT + 1;
          XX := XX + 8;
          case KEY of
               'C','c' : COMPLETE := 1;
               'P','p' : COMPLETE := 1;
               'Q','q' : COMPLETE := 1;
               'R','r' : COMPLETE := 1;
               'E','e' : COMPLETE := 1;
               'O','o' : COMPLETE := 1;
           end;
        end;
      end;  {case}
    end;
    get_choice := select;
end;
```

```
Procedure plott(var POINTERS : tarri351; MDIG,MSKEL : tarr51;
              var LCOUNT,FEEDARR : tar51; ELEMENT : tars51;
              SPACE,RAD,FEEDCNTR,SIZE : shortint);

begin
 GRAPHDRIVER := detect;
 initgraph(GRAPHDRIVER,GRAPHMODE,'');
 ERRORCODE := GRAPHRESULT;
 if ERRORCODE <> grOK then
 begin
      writeln('graphics error: ',grapherrormsg(ERRORCODE));
      writeln('program aborted');
      halt(1);
 end;
 circles(POINTERS,MSKEL,LCOUNT,RAD,SPACE);
 linedraw(MSKEL,POINTERS,SIZE,RAD,SPACE);
 outline(FEEDARR,FEEDCNTR,POINTERS,RAD,SPACE);
end;

Procedure graphtext(var POINTERS : tarri351; MDIG,MSKEL : tarr51;
                  LCOUNT,FEEDARR : tar51; ELEMENT : tars51;
                  SPACE,RAD,FEEDCNTR,SIZE : shortint);
begin
 PROBLEM := 0;
 while PROBLEM < 2 do begin
 settextstyle(2,horizdir,5);
 K := textwidth('A');
 L := textheight('A')+1;
 outtextxy(0,maxy-4*L,'P=Print  Q=Quit  R=Reduce  E=Enlarge  #=Description
                    O=Other graph  C=Compact');
 S := get_choice;
        PROBLEM := 0;
        setcolor(0);
        Y := MAXY-4*L;
        while Y <= MAXY do begin
              line(0,Y,MAXX,Y);
              inc(Y);
        end;
        setcolor(15);
        Y := round(MAXY - 3*L);
        I := 1;
        BOOL := S[1];
        case BOOL of
             'c','C' : begin
                         SPACE := round(0.75*SPACE);
                         PROBLEM := 1;
                         closegraph;
                         plott(POINTERS,MDIG,MSKEL,LCOUNT,FEEDARR,ELEMENT,
                           SPACE,RAD,FEEDCNTR,SIZE);
                       end;
             'e','E' : begin
                         SPACE := round(SPACE * 1.05);
                         RAD := round(RAD * 1.05);
                         TEXTSIZE := TEXTSIZE * 1.05;
                         CSIZE := round(TEXTSIZE);
                         PROBLEM := 1;
                         closegraph;
                         plott(POINTERS,MDIG,MSKEL,LCOUNT,FEEDARR,ELEMENT,
                           SPACE,RAD,FEEDCNTR,SIZE);
                       end;
             'r','R' : begin
                         SPACE := round(SPACE * 0.95);
                         RAD := round(RAD * 0.95);
                         TEXTSIZE := TEXTSIZE * 0.95;
```

```
                        CSIZE := round(TEXTSIZE);
                        PROBLEM := 1;
                        closegraph;
                        plott(POINTERS,MDIG,MSKEL,LCOUNT,FEEDARR,ELEMENT,
                          SPACE,RAD,FEEDCNTR,SIZE);
                      end;
            'o','O' : begin
                        closegraph;
                        PROBLEM := 2;
                      end;
            'q','Q' : PROBLEM := 3;
            'p','P' : begin
                        writeln('not ready yet');
                        PROBLEM := 1;
                      end;
        end;
        if PROBLEM = 0 then begin
           val(S, P, Q);
           str(P,S);
           while I < length(ELEMENT[P]) do begin
                 X := 0;
                 while X < MAXX do begin
                       outtextxy(X,Y,ELEMENT[P,I]);
                       X := X + K;
                       inc(I);
                       if I > length(ELEMENT[P]) then break;
                 end;
                 Y := Y + L;
           end;
        end;
 end;
 closegraph;
end;

procedure Abort(Msg : string);
begin
  Writeln(Msg, ': ', GraphErrorMsg(GraphResult));
  Halt(1);
end;
```

```
Procedure Digraph(MREACH : tarr51; ELEMENT : tars51; FEEDARR : tar51;
                  FEEDCNTR,MRSIZE : shortint);
var
   POINTERS : tarri351;
   LCOUNT       : tar51;

begin
  if RegisterBGIdriver(@EGAVGADriverProc) < 0 then
    Abort('EGA/VGA');
  if RegisterBGIdriver(@HercDriverProc) < 0 then
    Abort('Herc');
  if RegisterBGIfont(@SmallFontProc) < 0 then
    Abort('Small');
 PROBLEM := 1;
 CHOICE := 1;
 SIZE := MRSIZE;
 RAD := 15;
 SPACE := 24;
 clrscr;
 CSIZE := 5;
 TEXTSIZE := 5;
 for I := 0 to 50 do begin
    POINTERS[I,1] := 0;
    POINTERS[I,2] := 0;
    POINTERS[I,3] := 0;
 end;
 For I := 0 to SIZE do begin
     move(MREACH[I],MDIG[I],SIZE+1);
 end;
 while PROBLEM <> 3 do begin
       clrscr;
       thinking;
       blocklevel(MDIG,LCOUNT,CHOICE);
       Main(MDIG,MSKEL,SIZE,PROBLEM);
       feedin(MDIG,MSKEL,LCOUNT,FEEDARR,SIZE,FEEDCNTR);
       plott(POINTERS,MDIG,MSKEL,LCOUNT,FEEDARR,ELEMENT,SPACE,RAD,
            FEEDCNTR,SIZE);
       graphtext(POINTERS,MDIG,MSKEL,LCOUNT,FEEDARR,ELEMENT,SPACE,RAD,
                 FEEDCNTR,SIZE);
       SIZE := MRSIZE;
       if CHOICE = 1 then begin
          CHOICE := 2;
       end else begin
           CHOICE := 1;
       end;
  For I := 0 to SIZE do begin
     move(MREACH[I],MDIG[I],SIZE+1);
 end;
 end;
end;
end.
```

MODUNIT

```
Unit MODUNIT;
{Unit to Modify the MREACH array so that links can be added or removed
if so desired.}

Interface
 uses
     crt,typeunit;

 Procedure modify(var MREACH : tarr51; var ELEMENT : tars51;
                  var FEEDARR : tar51; var FEEDCNTR,MRSIZE : shortint);

Implementation

Var
   I,J,CHECKER           : shortint;
   K,L,M,N,P,Q,R         : integer;
   BOOL,CHOICE           : char;
   CONTEXT,RELATE        : string;

Procedure cutlink(var MREACH : tarr51; var FEEDARR : tar51;
                  var MRSIZE,FEEDCNTR : shortint);
{procedure to cut a link between two elements}
var
   ELEMENT1,ELEMENT2 : shortint;

begin
 Writeln('Please enter the element number FROM which you desire');
 Write('to remove a link : ');
 Readln(ELEMENT1);
 Write('Remove link from element ',element1,' to element number : ');
 Readln(ELEMENT2);
 Writeln('Are you sure that you wish to cut the link from element ',ELEMENT1);
 Write('to element ',ELEMENT2,' ? ');
 BOOL := Readkey;
 BOOL := upcase(BOOL);
 dellay;
 if BOOL = 'N' then exit;
 I := 1;
 If FEEDCNTR > 0 then begin
    M := 0;
    while I < FEEDCNTR do begin
          J := FEEDARR[I];
          for K:= 1 to J do begin
              inc(I);
              if ELEMENT1 = FEEDARR[I] then inc(M);
              if ELEMENT2 = FEEDARR[I] then inc(M);
          end;
    inc(I);
    end;
 end;
 If M > 0 then matexpand(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
 I := 1;
 while ELEMENT1 <> MREACH[0,I] do begin
       inc(I);
 end;
 J := 1;
 while ELEMENT2 <> MREACH[0,J] do begin
       inc(J);
 end;
 For K:= I to MRSIZE do begin
     if MREACH[K,I] = 1 then MREACH[K,J] := 0;
 end;
end;
```

```
Procedure addlink(var MREACH : tarr51; var FEEDARR : tar51;
                  var MRSIZE,FEEDCNTR : shortint);
{Procedure to add a link between two elements in the matrix}
var
   ELEMENT1,ELEMENT2 : shortint;

begin
 Writeln('Please enter the element number FROM which you desire');
 Write('to add a link : ');
 Readln(ELEMENT1);
 Write('Add link from element ',ELEMENT1,' to element number  : ');
 Readln(ELEMENT2);
 Writeln('Are you sure that you wish to add a link from element ',ELEMENT1);
 Write('to element ',ELEMENT2,' ? ');
 BOOL := Readkey;
 BOOL := upcase(BOOL);
 dellay;
 if BOOL = 'N' then exit;
 I := 1;
 If FEEDCNTR > 0 then begin
    M := 0;
    while I < FEEDCNTR do begin
          J := FEEDARR[I];
          for K:= 1 to J do begin
              inc(I);
              if ELEMENT1 = FEEDARR[I] then inc(M);
              if ELEMENT2 = FEEDARR[I] then inc(M);
          end;
    inc(I);
    end;
 end;
 If M > 0 then matexpand(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
 I := 1;
 while ELEMENT1 <> MREACH[0,I] do begin
       inc(I);
 end;
 J := 1;
 while ELEMENT2 <> MREACH[0,J] do begin
       inc(J);
 end;
 For K:= I to MRSIZE do begin
     if MREACH[K,I] = 1 then MREACH[K,J] := 1;
 end;
end;
```

```
Procedure delel(var MREACH : tarr51; var FEEDARR : tar51;
                var MRSIZE,FEEDCNTR : shortint);
{proceedure to diminish the size thi as questions are answered}
var
   ELEMENT1 : shortint;
Begin
 clrscr;
 Write('Please enter the number of the element you wish to delete : ');
 Readln(ELEMENT1);
 Writeln('Element ',ELEMENT1,' will be permanently remove from the model.');
 Write('Are you sure that you wish to delete element ',ELEMENT1,' ?');
 BOOL := Readkey;
 Writeln('');
 BOOL := upcase(BOOL);
 dellay;
 if BOOL = 'N' then exit;
 I := 1;
 If FEEDCNTR > 0 then begin
    M := 0;
    while I < FEEDCNTR do begin
          J := FEEDARR[I];
          for K := 1 to J do begin
              inc(I);
              if ELEMENT1 = FEEDARR[I] then inc(M);
          end;
    inc(I);
    end;
 end;
 If M > 0 then matexpand(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
 I := 1;
 while ELEMENT1 <> MREACH[0,I] do begin
       inc(I);
 end;
 for M := I+1 to MRSIZE do begin
     move(MREACH[M],MREACH[M-1],MRSIZE+1);
     for K := 0 to (MRSIZE) do begin
         MREACH[K,M-1] := MREACH[K,M];
     end;
 end;
dec(MRSIZE);
end;
```

```
Procedure addel(var MREACH : tarr51; var FEEDARR : tar51;
                var MRSIZE,FEEDCNTR : shortint; var ELEMENT : tars51);
{Procedure to add elements to the  element set, so that they can be
modelled with those already entered}
begin
 matexpand(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
 J := 1;
 while J < 50 do begin
       clrscr;
       L := 0;
       for I := 1 to MRSIZE do begin
           if J = MREACH[I,0] then inc(L);
       end;
       if L = 0 then begin
          Writeln('Please enter the description for element ',J);
          Writeln('(Enter ''X'' to exit.)');
          Readln(ELEMENT[J]);
          if ELEMENT[J] = 'x' then begin
             matreduce(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
             exit;
          end;
          if ELEMENT[J] = 'X' then begin
             matreduce(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
             exit;
          end;
          inc(MRSIZE);
          for I := 1 to MRSIZE do begin
              MREACH[MRSIZE,I] := 2;
              MREACH[I,MRSIZE] := 2;
          end;
          MREACH[MRSIZE,0] := J;
          MREACH[0,MRSIZE] := J;
          MREACH[MRSIZE,MRSIZE] := 1;
       end;
 inc(J);
 end;
 Writeln('I''m afraid that the program is limited to modelling 50 elements,
           and');
 Writeln('there is no room for any more elements.');
 Writeln;
 writeln;
 Writeln('Press any key to continue');
 Readkey;
 dellay;
end;
```

```
Procedure model(var ELEMENT : tars51);
{Procedure to modify the description of an existing element.}
begin
BOOL := ' ';
 while BOOL <> 'N' do begin
 clrscr;
 Write('Please type the number of the element you would like to modify.');
 Readln(J);
 Writeln;
 Writeln;
 Writeln('Element number ',J,' current reads as follows :');
 Writeln;
 Writeln(ELEMENT[J]);
 Writeln;
 Writeln('Please enter new description of element ',J);
 writeln;
 Readln(ELEMENT[J]);
 Writeln;
 Writeln;
 Write('Would you like to modify another element (Y/N)');
 BOOL := Readyn;
 end;
end;

Procedure modify(var MREACH : tarr51; var ELEMENT : tars51;
                 var FEEDARR : tar51; var FEEDCNTR,MRSIZE : shortint);
begin
CHOICE := ' ';
While CHOICE <> 'X' do begin
 clrscr;
 writeln;
 writeln;
 writeln;
 Writeln('You are now in the link modification part of the modelling.');
 Writeln;
 Writeln('(A) Add a link between two elements.');
 Writeln('(B) Remove a link between two elements.');
 Writeln('(C) Delete an existing element.');
 Writeln('(D) Add a new element.');
 Writeln('(E) Modify an existing element''s description.');
 writeln;
 Writeln('(X) Exit');
 writeln;
 Writeln('Please enter ''A'',''B'',''C'',''D'',''E'' or ''X''. ');
 CHOICE := readkey;
 CHOICE := Upcase(CHOICE);
 dellay;
 clrscr;
 Case CHOICE of
     'A' : addlink(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
     'B' : cutlink(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
     'C' : delel(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
     'D' : addel(MREACH,FEEDARR,MRSIZE,FEEDCNTR,ELEMENT);
     'E' : model(ELEMENT);
 end;
end;
matexpand(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
matreduce(MREACH,FEEDARR,MRSIZE,FEEDCNTR);
end;
end.
```

DISKUNIT

```
Unit DISKUNIT;
{Unit to Save and retrieve the MREACH array so that data can be stored
on disk for posterity.}

Interface

 uses
     crt,typeunit;

 Procedure disk(var MREACH : tarr51; var ELEMENT : tars51;
                var FEEDARR: tar51; var FEEDCNTR,MRSIZE : shortint;
                var CONTEXT,RELATE : string; BQ : tar3; BN : tar2);

Implementation

Var
   I,J,CHECKER           : shortint;
   K,L,M,N,P,Q,R         : integer;
   JUNK,BOOL,CHOICE      : char;
   F                     : text;
   T                     : string;

Procedure savefil(MREACH : tarr51; CONTEXT,RELATE : string; ELEMENT : tars51;
                  MRSIZE : shortint; BQ : tar3; BN : tar2);
{Procedure to save file to disk.}
begin
 clrscr;
 writeln('Please enter the filename under which you would like the model to
         be stored.');
 write('Please note that a ''.ISM'' suffix will be added to the filename.');
 readln(T);
{make sure t is only 9 chars long}
 if length(T) >9 then T[0] := #9;
 T:=T+'.ism';
 assign(F,T);
 rewrite(F);
 writeln(F,MRSIZE);
 writeln(F,CONTEXT);
 writeln(F,RELATE);
 for I := 1 to MRSIZE do begin
     writeln(F,ELEMENT[I]);
 end;
 for I := 0 to MRSIZE do begin
     for J := 0 to MRSIZE do begin
         CHECKER := MREACH[I,J];
         write(F,CHECKER,' ');
     end;
     writeln(F);
 end;
 for I := 1 to 4 do begin
     writeln(F,BQ[I]);
 end;
 for I := 1 to 3 do begin
     writeln(F,BN[I]);
 end;
 close(F);
end;
```

```
Procedure recovfil(var MREACH : tarr51; var CONTEXT,RELATE : string; var
                   ELEMENT : tars51; var MRSIZE : shortint);
{Procedure to recover file from disk.}
begin
 clrscr;
 writeln('Please enter the filename of the model to be retrieved.');
 write('Please note that a ''.ISM'' suffix will be added to the filename.');
 readln(T);
{make sure t is only 9 chars long}
 if length(T) >9 then T[0] := #9;
 T:=T+'.ism';
 assign(F,T);
 reset(F);
 readln(F,MRSIZE);
 readln(F,CONTEXT);
 readln(F,RELATE);
 for I := 1 to mrsize do begin
     readln(F,ELEMENT[I]);
 end;
 for I := 0 to MRSIZE do begin
     for J := 0 to MRSIZE do begin
         read(F,CHECKER);
         mreach[I,J] := CHECKER;
     end;
     readln(F);
 end;
 close(F);
end;
```

```
{Main driving procedure}
Procedure disk(var MREACH : tarr51; var ELEMENT : tars51;
               var FEEDARR: tar51; var FEEDCNTR,MRSIZE : shortint;
               var CONTEXT,RELATE : string; BQ : tar3; BN : tar2);
begin
 JUNK := ' ';
 while JUNK <> 'X' do begin
       clrscr;
       writeln;
       writeln;
       writeln;
       Writeln('Please enter the letter corresponding to your choice:');
       Writeln;
       Writeln('(A) : Save model to disk.');
       Writeln('(B) : Retrieve saved model from disk.');
       Writeln;
       writeln('(X) : EXIT.');
       Writeln;
       Writeln('Please enter ''A'',''B'', or ''X''. ');
       JUNK := readkey;
       JUNK := Upcase(JUNK);
       dellay;
       Case JUNK of
       'A' : begin
                Matexpand(MREACH,feedarr,MRSIZE,feedcntr);
                savefil(MREACH,CONTEXT,RELATE,ELEMENT,MRSIZE,BQ,BN);
                matreduce(MREACH,feedarr,MRSIZE,feedcntr);
           end;
       'B' : begin
                if MRSIZE > 0 then begin
                   clrscr;
                   writeln('        WARNING : CURRENT MODEL WILL BE
                         PERMANENTLY LOST.');
                   writeln;
                   write('Are you sure that you wish to load a new model ? ');
                   BOOL := readyn;
                   dellay;
                   if BOOL = 'Y' then begin
                      recovfil(MREACH,CONTEXT,RELATE,ELEMENT,MRSIZE);
                      matreduce(MREACH,feedarr,MRSIZE,feedcntr);
                   end;
                end else begin
                recovfil(MREACH,CONTEXT,RELATE,ELEMENT,MRSIZE);
                matreduce(MREACH,feedarr,MRSIZE,feedcntr);
                end;
           end;
       end;
 end;
end;
end.
```