Data Capture of Geometric Data for Local Authorities' Geographic Information Systems

by

Francois Johan de Wet

Thesis presented in partial fulfilment of the requirements for

the degree of

Master of Science in Engineering (M.Sc.(Eng.))

Department of Surveying and Geodetic Engineering

University of Cape Town.

25 May 1995.

Acknowledgements.

The author records his sincere thanks and appreciation to the following:

Mr M B Barry of the Department of Surveying and Geodetic Engineering for his guidance and assistance during the course and during the preparation of this thesis.

Messrs Cadastral Computing Co. (Pty) Ltd for valuable comments and suggestions during the development and testing of the programs for the capture of cadastral data.

Mr Michael Fielding and Mrs Jeannette van Zyl for the proof reading of this dissertation.

Certification.

This thesis is submitted as a "half thesis" in partial fulfilment of the requirements for the degree M.Sc.(Eng.). I certify that this work is my own and, where applicable, that I have acknowledged the work of other persons who have contributed to this thesis. This includes all the computer programs, procedures and algorithms used in the Data Handling System (the RDS) and the Data Capture Programs described in this thesis.

F J de Wet.

25 May 1995.

SYNOPSIS

Introduction.

This thesis describes research and development work which led to algorithms, procedures and computer programs which facilitate the cost effective and accurate capture of geometric data. The geometric data for a Geographical Information System (GIS) at a local authority or municipality consist of a number of different data sets. These include inter alia: the cadastral information, zoning information, servitudes, building lines, the outlines of improvements and the reticulation networks and the house connection points of the engineering services.

The initial capture of the geometric data appears to be deceptively simple and is often not given the required consideration. The initial data capture phase of GIS projects is usually a difficult and time consuming process. This is even more so in the case of GIS for local authorities. The reason for this difficulty is the large volume of data coupled with the high accuracies required for the cadastral base map and the engineering services. Input facilities of most commercial GIS software packages generally do not provide the most efficient means of data capture. This problem warrants the development of techniques and procedures specific to local authority GIS applications which ensure that data capture can be done effectively and efficiently.

The major benefit of these procedures is that they can be implemented on personal computers with low random access memory capacity. This eliminates the need for investment in costly equipment at the initial stage of data capture in the development of a GIS. It allows the capture of data on low cost technology and the postponement of the purchase of an expensive system or workstation until the data capture phase has been completed. The lowest personnel skills required are copy typing in contrast to the traditional methods of using CAD operators who command higher salaries and require more expensive training. The system developed by the author is more productive, both in quality and volume of work produced, than the CAD approach. It also permits the delay of purchase and training on expensive GIS software and hardware, which may be obsolete by the time the graphic data base is established.

A Data Handling System for a Large Volume of Data.

An important aspect of the procedures and programs for data capture is the need for an efficient data handling scheme for the manipulation of large volumes of data. A system for the storage and retrieval of data has been developed which simplifies the manipulation of large volumes of data, even on personal computers with limited storage capacity. This is one of the major advantages of the methodology described in this thesis.

Data are stored in random access files and these files reside on the hard disk drive at all times. An area in random access memory (RAM) is reserved on which portions of the random access files are mapped, thereby enabling the data to be accessed directly from RAM. When data are required which reside only on disk, a paging algorithm will automatically swop a portion of the data in RAM with a portion of the random access file to make the required data accessible in memory. This page-swopping technique effectively creates a virtual memory containing the whole random access file or volume of data.

The procedures in the system furthermore provide various tools and utilities. These are used for the storing and retrieving of data in a number of convenient ways and for implementing the various required data structures.

The Graphics Data Structure.

Physical features on and under the ground are represented in a GIS as graphic entities in the geometric data base. Data capture procedures must collect all the necessary data to develop the full intelligence of the GIS information.

The graphic entities which represent the physical features in the data base are points, lines and polygons. Graphic entities which represent the cadastral layout are called the primary entities. The cadastral layout forms the base map of a municipal GIS and consists of one or more outside figures which contain the blocks and street reserves. These blocks are divided into the individual lots or land parcels.

The remaining geometrical entities are called the secondary graphic entities. Secondary polygons represent servitudes or the outlines of buildings that overlay the land parcels. Secondary lines represent the pipes or cables of the water, sewer or electrical reticulation networks that cross lot and block boundaries or other services. The connectivity between the

primary and secondary entities can be generated; i.e. the lots that are overlaid by a servitude or the lots that are traversed by a pipeline etc.

Data Capture of Cadastral Information.

The most accurate sources of data for the cadastral base map of a local authority GIS are the land surveyors' documents; i.e. general plans and diagrams, stored in the offices of the Surveyors General. A methodology has been developed for the efficient and accurate capture of this information.

All the information from the source documents is gathered through the use of a data typing program developed by the author. Quality control checks on data consistency are performed to eliminate typing errors as far as possible. A number of features are incorporated in the typing program which limit the required number of keystrokes. The source data are then processed to generate all the additional geometric information for the construction of a continuous map. The final step comprises the generation of the graphic entities, the development of the topology and the classification of the entities. A plot of the cadastral layout can be generated for visual inspection. The data capture procedure is independent of any particular GIS software or hardware and the data are stored in a local data base. Data verification calculations and checks on tolerances are performed during the whole data capture process.

The data are exported in export file formats as determined by the target GIS. This procedure requires export format interfaces which have been developed for a number of GIS systems. Similar additional interfaces can be created according to new specifications if required.

A number of procedures were also developed to extract information from other data sources such as CAD drawings. Calculations are performed to reconstruct the individual lot information, topology and data classification for the whole cadastral layout.

Data Capture of Engineering Services.

The preferred sources of data for the engineering services are as-built or design drawings. Procedures have been developed for the capture of this data. There are a number of similarities between the procedures for data capture of cadastral data and engineering services data.

The required information is entered using a dedicated data typing computer program. A number of checks are performed to eliminate obvious typing errors. Additional calculations are performed and the geometry of the particular service network is constructed. The final step comprises the generation of the graphic entities, the development of the topology and the classification of the entities.

As in the previous case, the data are stored in the local data base. The procedures are, therefore, independent of any particular commercial GIS. Data are exported in file formats as determined by the target commercial GIS and this again requires export interfaces as described in the section relating to cadastral data capture.

Special digitising programs for the horizontal layouts of the reticulation networks were developed. Various techniques have been employed to make these programs efficient, less dependent on the skills of the operator and to enhance their accuracy.

Data Capture of Buildings and Improvements

The data capture for improvements primarily involves the digitising of the positions and outlines of the buildings from the valuation sheets or building plans. This is done by using a dedicated digitising program in which a number of techniques are employed to improve efficiency and accuracy and to limit the dependency on operator skills.

A Low Cost GIS for Local Authorities.

Many of the program modules developed for the data capture system have been incorporated into a small and simple but effective system for the storage and retrieval of the cadastral data, engineering services data, building positions, valuation data and land owner's information. This can provide a low cost information system for the storage and retrieval of data which are related to individual land parcels.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS.

GLOSSARY.

Address. Identification for the storage position or location of a data record or unit of data.

Application model. Data base description of the real word under consideration in a GIS.

As-built drawing. Engineering drawings indicating the extent of constructions as they were built.

ASCII. American Standards Code for Information Interchange. Standard code for characters as manipulated and displayed by computers.

Atom. The elementary building block of data structures. An atom corresponds to a record in a file and may contain one or more fields of data. Also called a node.

Auxiliary memory. Data storage other than main memory; for example, storage on hard disk drive.

Average. The statistical mean; the mean value.

Base map. Basic reference frame for a geometric model.

Binary tree. A tree in which each node has at most two outdegrees, i.e. edges leaving a node.

Bit. Either zero or one. It is derived from *bi*nary digi*t.*

Block. (Cadastral) Tract of land bounded by streets and encompassing individual land parcels.

Block. (Data) The unit of data which can be retrieved directly from the random access file.

Block size. The size of the directly accessible data block of a random access file, usually expressed in terms of the number of words or bytes

Bresenham. Author of algorithms for raster representation of lines and circles, which bear his name.

Byte. Smallest addressable unit of storage on a computer and a collection of eight bits.

Cache. Special temporary storage buffers in fast access memory to speed up data i/o's of slower storage devices.

CAD. Computer Aided Design

Cadastral. The extent of land for purposes of ownership, valuation and taxation.

Clustering. A build-up of table entries around a single table entry or along a pattern in a hashing function. This is opposite to an even or random distribution.

Collision. An act that occurs when two or more keys hash to the same address.

Consistency. State of being compatible and not contradictory.

Continuous map. A seamless map covering the total area under consideration and not limited to a sheet size.

Cross-referenced list. A double linked list specifying the relations and inverse relations between two sets of entities.

Data acquisition. The activity of collecting relevant data for storage in a data base.

Data capture. See data acquisition

Data field. A unit of information.

Data block. The unit of data stored or retrieved from the random access file.

Data book. A logical collection of data blocks or sub-set of a random access file.

Data bucket. See hash bucket.

Data entry. The unit of data on the data pages. Corresponds to a record.

Data file. A collection of related records treated and stored as a unit.

Data header. See header.

Data index. See index.

Data link. Pointer to the next data record in a sequence of records or list.

Data page. The unit of related data from a data book that corresponds to a block of the random access file.

Data pointer. See pointer.

Data record. See record.

Data set. A collection of data for a project stored in one or more files.

Data handling scheme. A methodology and related procedures for the storage and retrieval of data.

Data structure. The organisation or arrangement of the data units in a data collection.

Digitise. Converting positional information on a drawing or diagram to digital co-ordinated information by means of special peripheral equipment.

DXF. The de facto standard Autocad exchange format for drawing files.

Edge. An edge connects two nodes in a graph. An edge may or may not have direction.

Engineering Service. Public utility service provided to land parcels.

Erven. A number of lots or land parcels.

Export. To write information to a file in a prescribed format for further processing.

FIFO. First in, first out queue discipline.

Forward pointer. A pointer that tells the location of the next item in a data structure. It corresponds to a directed edge in a graph.

Geometric data base. Structured collection of geometric related data.

Geometric model. Geometric description of the real word under consideration in a GIS.

Graph. A set containing two types of objects: nodes and edges. These provide a mathematical model for data structures in which the nodes correspond to data items, and the edges to pointer fields.

Graphic entity. Smallest graphical unit that can be manipulated individually.

Graphic item. See graphic entity.

Graphic object. A collection of graphical entities or items that can be manipulated collectively.

Graphic primitives. Functions used to specify graphic entities to make up the graphic picture.

Grid. Rectangular pattern or arrangement of lines forming a matrix of squares.

Grid cell. The unit of a grid or raster.

Hashing. A key-to-address transformation in which the key determines the location of the data field.

Hash algorithm. Specific rules for hashing

Hash function. Procedure for the execution of key-to-address transformation.

Hash number. Address returned by hash function.

Header. A special data item that points to the beginning of a list.

Helmert transformation. A least-squares co-ordinate transformation for an overdetermined system of equations.

Import. To read information from a file in a prescribed format for further processing.

Incidence matrix. A two-dimensional array which describes the edges in a graph. Also called a connection matrix.

Indegree. The number of directed edges which point to a node.

Index. A symbol or numeral which locates the position of an item in an array.

Intelligence. Additional descriptions and relationships which transform the geometric data into an application model.

Interface. A software procedure to translate data from one format to another.

Join. The distance and bearing between two points.

Key. One or more fields in a record that are used to locate the record.

Key to address. See hashing.

Leaf. A terminal node of a tree.

Label. A name or identification for a node or record. Can be used as a key.

Land parcel. Piece of land. See lot.

Level. A measure of the distance from a node to the root of a tree.

LIFO. Last in, first out stack discipline.

Linear search. To accomplish a linear search, begin with the first element and compare until matching key is found or the end of the list is reached.

Linked list. A list in which each atom contains a pointer to the location of the next atom.

List. An ordered collection of atoms.

Lot. Plot or allotment of land.

Map, page map. Temporary copy of permanently stored data, usually in fast access memory for fast and easy access.

Model. Structured data description and relationships relevant to real world application under consideration.

Multilinked list. A list in which each atom has two or more pointers.

Nil pointer. A pointer used to denote the end of a linked list.

Node. See atom.

Network. Collection of links connecting nodes for the provision or reticulation of services, i.e. water reticulation network.

Outside figure. The boundaries describing the extent of a township.

Paging. The activity of copying a block of data from RAM to disk in order to create space for a block of data to be copied from disk to RAM.

Paging algorithm. The rule whereby it is determined which one of a number of data blocks are to be used for page swapping.

Paging function. The procedure that implements the paging algorithm.

Pixel. The smallest picture unit or dot accepted by point-plotting display devices.

Pointer. An address or other indication of location.

Polygon. Plane rectilinear figure with many angles or sides.

Pop. The act of removing an element from a stack. Also called pull.

Pull. See pop.

Push. The act of placing an element on a stack. Also called put.

Put. See push.

Probing. Searching or exploring from a starting point along a calculated path for a required entity.

Quad tree. A tree in which each node has an outdegree of four.

Queue. A list that allows insertion of atoms at one end and deletion of atoms at the opposite end.

RAM. See Random access memory.

Random access. A method of retrieving data from a storage area in which the retrieval time is independent of the location of the data. Contrast with sequential access.

Random access file. A data storage file on the hard disk drive from which data can be retrieved directly and independently of the location of the data.

Random access memory. The central storage area from which data can be retrieved directly and independently of the location of the data.. Also known as RAM

Raster. A rectangular grid or matrix of cells.

Record. A collection of related data items. A collection of related records makes up a file.

Referenced list. A linked list specifying the relations between two sets of entities.

Remainder. Portion of original land parcel left over after division.

Reticulation. Distribution of a service to individual users.

Road reserve. Corridors of land set apart for the construction of roads.

Root. The node with indegree zero.

Scan. An algorithmic procedure for visiting or listing each node of a data structure.

Scatter storage. See hashing.

Sequential data structure. A data structure in which each atom is immediately adjacent to the next atom. Also called contiguous data structure.

Servitude. Subjection of tenement to an easement. Right-of-way over a piece of land held by an owner.

Stack. A list that restricts insertions and deletions to one end.

Stand. Piece of land. See lot, land parcel.

Structure. The organisation or arrangement of the parts of an entity.

Subdivision. Portion of land cut off from a land parcel during division. As opposed to remainder.

Swop, page swopping. See paging.

Topology. Connectivity of components giving a qualitative description of the shapes and relations between objects and entities.

Township. Site laid out for a town, small town or a territorial division of a larger one.

Tree. A structure which has a unique node , the root, whose successor set consists of all the other nodes.

Virtual memory. Fictitious or simulated random access storage area which is greater than physical RAM.

CHAPTER 1

Introduction.

1.1 Introduction.

The application of a Geographic Information System in a local authority or municipality is concerned mainly with the information relating to land use and the services infrastructure. It is often called a Land Information System (LIS). In general, the initial data capture for a GIS is a major part of the implementation phase. It takes up an even larger portion of the budget in the case of a local authority GIS. The main reasons for this are the extent of the data base, the volume of the data and the required accuracy of the data for the cadastral base map and the engineering services. It is falsely believed that data stored in existing digital formats by local authorities, or which may be obtained from consultants or other parties, will always lead to substantial savings in cost. These data sets frequently require extensive processing before they are acceptable for incorporation into the GIS. The geometric data in these CAD, text, spreadsheet or data base files are usually not topologically structured. It is also preferable to have control over the accuracy of the data capture and to be able to verify the accuracy of the source documents. In practice, information on this is seldom stored in data dictionaries or similar documents. Therefore, in the author's opinion, quality checks should be done on all data, whether acquired from existing digital sources or paper documents.

The initial data capture of the cadastral layout and engineering services requires a substantial investment in terms of time and money. It therefore constitutes a substantial portion of the direct cost of the start-up phase of the GIS. The methods developed in this thesis reduce both the time and cost and also the risk of failure.

1.2 The Start-up Phase of a GIS.

The expenditure in establishing and running a GIS can be divided into four main categories. The first and obvious category is the purchase price of the system. This is the total procurement cost of the equipment, which consists of the hardware and software, and includes the cost of delivery, installation and formal training. This expenditure is a direct cost.

The second category is the establishment of the infrastructure required to design, implement, operate, maintain and support the GIS. This portion of the expenditure is considered to be an indirect cost. This cost includes the costs of system planning, system design, system start-up, system commissioning, system testing, the appointment and training of maintenance and support personnel, and general familiarisation with the system. It is of vital importance that this phase of the establishment of a GIS is managed correctly to ensure the success of the system. Time and money spent wisely during this phase will ultimately lead to substantial savings in the long run. Although not included in the scope of this dissertation, it should be mentioned that this phase of the establishment of a GIS is often ignored or underestimated during system planning. (De Wet, 1993)

The third category, which is a direct cost and is part of the start-up phase, is the initial data capture. An information system can only operate effectively and efficiently if the information is as comprehensive, complete and accurate as required by its users. It is also desirable to get the GIS up and running in the shortest possible time in order to get a return on investment as soon as possible. The execution of the initial data capture phase in a short time can be a task of considerable magnitude, and can represent a substantial portion of the GIS start-up costs.

The fourth and last category is the ongoing expenditure for the normal operation, system support, user support, maintenance, updating and upgrading of the system.

1.3 The Initial Data Capture.

The initial data capture of the geometric data looks deceptively simple, and is often not given the required consideration. The author is aware of a number of cases where the initial budget only provides for the hardware and software. In these cases very little attention was given to the planning and design of the system. An approach is often adopted that the data capture can be done by the technical or administrative staff during slack periods between their normal duties. This approach is likely to fail. In a number of cases the development of a GIS has not progressed much further than a CAD layout of the cadastral base map after two or three years. This is due to insufficient planning and management of the data capture phase.

A number of papers can be found in the literature dealing with various aspects of establishing a geographic information system. (Prinsloo 1991; De Wet, 1991; Dickenson and Calkins, 1988; Labuschagne, 1993; Macdevette, 1991; Van Rensburg, 1990; Van Rensburg and Dickenson, 1991) These aspects range from practical considerations to feasibility studies and cost benefit analyses. Although some of these papers refer to the initial data capture exercise, there is little in the literature which deals with this aspect in great detail, particularly for a local authority application. The author was unable to find literature that indicates the relative proportion of the cost of initial data capture to the overall project start-up cost.

The cost of the initial data capture effort depends on many factors. Some of these are the extent of the data base, the volume of data to be captured, the method of data capture and the required data accuracy and tolerances. According to Smith and Tomlinson (1992) investigations at a number of GIS sites have produced figures of between 33% and 66% of the GIS start-up cost being spent on the initial data capture phase. The author believes that a figure of as low as 33% represents cases of a limited database, low accuracy requirements and bulk importation of data from external sources with limited processing involved in getting the data into a suitable format. Figures have been quoted of up to 80% of the budget for the establishment of a GIS being spent on the data capture phase. (Anon, 1989). A figure of 60% seems to be more in line with the majority of local authority GIS's.

In a paper based on a survey of 43 medium sized local authorities (towns with 7000 to 10000 lots), de Villiers (1994) describes the main problem areas experienced during the implementation of a GIS as implementation, budget and data input and transcription. The transfer of data from plans to digital information is further singled out as the most time and cost consuming aspect of getting a GIS into operation. According to the investigation, these medium sized municipalities require on average 4,19 man years to complete the initial data capture phase of the cadastral and engineering services data.

The following three examples, based on actual observations, will illustrate the relative size of the cost components of implementing a GIS in a local authority. The prices are based on current (1994) prices from five actual tenders and quotations for similar situations. The prices are the rates that were accepted by the clients and were offered by third party tenderers and not the author. Consider the probable expenditure for GIS

projects for three municipalities with approximately 3500, 8000 and 15000 land parcels respectively. The costs for the system, design and data capture were as follow:

No of Lots	**3 500**	**8 000**	**15 000**
System (Hardware and Software)	R 32.000,00	R 58.000,00	R180.000,00
Data base design and system set-up	R 28.000,00	R 30.000,00	R100.000,00
Data capture	R 91.000,00	R128.000,00	R450.000,00
Total	**R151.000,00**	**R216.000,00**	**R730.000,00**

The data capture portion represents 60,3%, 59,2% and 61,6% of the total cost for the project set-up phase respectively.

The data capture portion yields a tariff of R26,00, R16,00 and R30,00 per lot respectively. These figures require further explanation. The tariff of R16,00 per lot was made up of R3,00 per land parcel for the cadastral data and R13,00 per land parcel for the engineering services data. The R3,00 per lot for cadastral data seems to be the lower limit of the current going rate for this type of work when data capture is done from general plans. If the data for a substantial number of the land parcels are to be obtained from diagrams, as opposed to general plans, the tariff can be as high as R12,00 per lot. This will push the average tariff up to between R4,00 and R5,00 per lot. The tariff of R13,00 per lot for engineering services data corresponds to the going rate for data detail up to the reticulation level. If these data have to be captured up to the detail level of house connections for water, sewer and electricity, up to the level of street entrances and street furniture and also zoning data, building lines and outlines of improvements, the tariff can be as high as R30,00 or more per lot.

Furthermore, these tariffs are for providing the data in an import file format so that additional costs will be incurred to load and test the data. It is therefore not unreasonable to have a potential initial data capture cost of more than R40,00 per lot, which will increase the total expenditures. This can also increase the relative portion of the data capture phase to well above 60% of the cost of the set-up phase.

The quality of the data depends on a number of factors. It is of vital importance that comprehensive specifications should be developed for the data capture irrespective of whether this is to be done in-house or subcontracted to third parties. This aspect is often ignored. The author has seen specifications for data capture in tender documents which

merely state that data capture is required for cadastral information and engineering services. This also severely limits the control over the data received and inevitably leads to substantial effort and cost to load, test and verify the data. Big savings in terms of time and cost can be realised if comprehensive specifications are drawn up in which responsibility for completeness and accuracy can be placed on the contractor and which will lead to full control over the data and minimal effort and cost for data verification.

Obviously the hardware costs can escalate if additional computers, network equipment and plotters are considered, but then again so also will the costs for data capture if the number of lots exceed those given in each example. The above arguments and examples emphasise that the cost of the initial data capture constitutes a substantial part of the initial start-up cost and therefore warrants careful consideration.

1.4 Special Data Capture Techniques.

The normal input facilities of commercial GIS's are not the most efficient tools for data capture. Most systems also provide facilities for the importation of data via formatted import files or the de-facto standard file formats of popular CAD, data base or spreadsheet programs, such as DXF, dBASE, Lotus and comma delimited ASCII files. However, obtaining data on a magnetic medium and importing the files into a GIS does not constitute data capture. Such digital data sets are seldom topologically structured and the data attributes of each graphic entity are often not incorporated in the digital data. A DXF file, for example, is an ASCII file that contains drawing information about lines, symbols, text and other drawing entities. This is not intelligent GIS data. For example, it requires a considerable amount of additional work and processing before the necessary intelligence is generated for a lot to be identified by its number. Additional information like area, zoning, owner, valuation, building restrictions, position of sewer, water and electrical connections, street address, street entrance, adjacent lot numbers also have to be added. Similarly, the lines, symbols and text in a drawing file of a sewer reticulation network require a considerable amount of work before the necessary intelligence has been added to provide information regarding size, material, depth, levels, grades etc. or to enable the system to be queried for the lot numbers that are serviced by a particular network segment, lots that are traversed by a pipe or cable, the points of intersection of the reticulation networks of the various services etc.

Special techniques and procedures developed in the author's methodology provide the required GIS information with the full topology and intelligence at little or no extra effort and cost than would be required to develop a CAD drawing of the cadastral or reticulation layout. Therefore the use of specialised data capture techniques is indispensable if the data capture is to be done effectively and within reasonable cost.

These data capture procedures should be:

- Fast, efficient and accurate.
- Automated for speed and operator independence.
- Capable of being used by semi-skilled operators.
- Optimised to use the fewest keystrokes.
- Capable of execution on inexpensive personal computers

From a theoretical point, many aspects of a fully fledged GIS are incorporated in the data capture system. These include data structures, data bases, data storage and retrieval, data sorting and searching and reporting as well as techniques and algorithms for computer graphics and data bases.

1.5 Scope of Work.

The remainder of this thesis describes research and development work which created algorithms, procedures and computer programs that facilitate the cost effective and accurate capture of geometric data. The geometric data for a local authority or municipal geographical information system consist of a number of different data sets. These include inter alia: the cadastral information, zoning information, servitudes, building lines, outlines of improvements, the reticulation networks of the engineering services and connection points to the engineering services.

The remaining chapters are arranged in three major parts:

The first part, consisting of chapters 2 and 3, describes in detail the techniques, procedures and data structures that were developed for the storage, retrieval and manipulation of large data sets on small computers which have limited processing power and memory size.

Part two, consisting of chapters 4 and 5, presents a description of the data capture programmes and the various techniques employed for fast, accurate and efficient data capture. Particular attention is given to control over the quality of the data through the monitoring of tolerances and accuracy.

Part three, consisting of chapters 6, 7 and 8, examines the application of the procedures, techniques and programs and describes a number of learning points gained from practical experiences.

Details of the algorithms, procedures, programs and examples are given in the appendices.

CHAPTER 2

A Data Handling System for Large Volumes of Data.

2.1 Introduction.

An important requirement of the procedures and programs that are described in the following chapters, is the availability of an efficient data storage and retrieval scheme. Reasons for this data handling system are that:

- Large volumes of data have to be manipulated.
- The data should be manipulated on personal computers with limitations in memory size.
- There should be flexibility for maintenance and extensions to the software procedures.

A data handling scheme, called the **Record Data System** or **RDS**, has been developed to manipulate very large volumes of data on small computers. The development of the RDS was inspired by a paper by Bettis (1977) when the author was involved in the development of pre-processors and post-processors for Finite Element Analysis programs in the early 1980's. These were implemented on mini computers with 16 bit operating systems and had to run in the limited address space of 64 kilobytes. The development evolved over a number of years and the system has proved to be invaluable in a number of programming projects since then; even when using computers with much larger address spaces. It is not uncommon on large data capture projects to exceed the 640 kilobyte memory limit of IBM compatible personal computers when using the standard data structures and arrays of the programming language.

The objectives of the RDS are:

- Optimum use of random access memory and external storage is to be achieved.
- Data with a complex composition must be handled in a standard and orderly procedure.
- A proven standardised set of procedures must be provided that can easily be incorporated into a programming project and which can be used with confidence.

2.2 Overview of the Record Data System.

A description of the data hierarchy and some terminology in this section provides a synopsis of the system and is followed by a more detailed discussion in the remainder of this chapter.

The unit of data that can be manipulated, that is stored and retrieved, is called a **data record** and hence the name **Record Data System** or **RDS**. A data record consists of a number of **data fields**. These data fields are typically numbers or text (character strings). The data records are stored in one or more random access files.

A volume of data is called a **data set** and is stored in a random access file. The random access file is logically declared as consisting of a number of **books** and each book consists of a number of **pages**. The data pages have a fixed size or length. The data pages contain the data **records** and each record is a collection of a number of data **fields**. The data records are the smallest data entities that can be stored or retrieved. The whole record has to be retrieved in order to access a specific data field, i.e. when a specific data field is required or has to be updated. This is shown schematically in Figure 2.1.

The data records are accessed via a system of pointers, indexes and reference tables. These pointers, indexes and reference tables are stored in special data pages called **pointer pages**.

The number and size of the data fields in a record, and therefore the size of a data record, are fixed for a particular book. The data record size can vary for different books and therefore the number of records per page will vary accordingly for the various books. The number of pages per book and the number of books per data set is arbitrary and open ended. This is theoretically restricted only by hardware constraints but in practice by some practical considerations. These practical restrictions, however, are seldom a problem. For example, the files, which can grow dynamically, can become large and this may be a problem on computers with limited available storage space. In addition, using two-byte words for pointers will restrict the number of records in any book, index, pointer or reference table to 32767. Four-byte pointers should be used when this limit is exceeded.

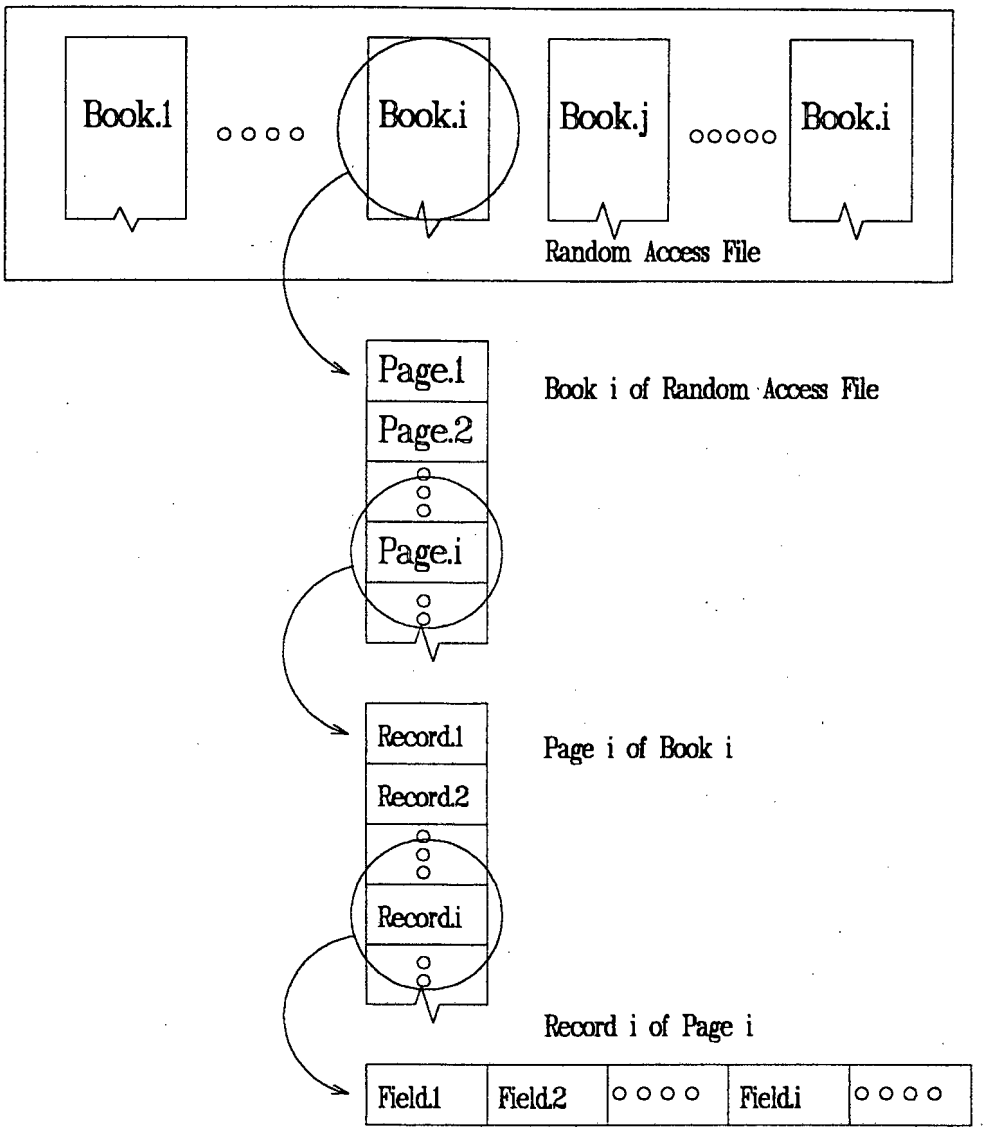


Figure 2.1 : Schematic Outline of Data Hierarchy of the RDS.

The data pages reside on the **random access file** at all times. The data pages are of fixed size or length. This page size must be a multiple of the physical block size of the random access file, i.e. the unit of data that can be retrieved directly from the random access file. The data pages will occupy specific data blocks in the random access file. An area in Random Access Memory, called the **memory map**, is reserved to keep copies of, or to map, a number of data pages of the data set. The memory map, therefore, contain copies of the corresponding blocks of the data file. A table of pointers, called the **map index**, is

maintained to keep track of the pages currently in memory. The map index therefore also maintains the relation between the logical data pages of the data set and the physical data blocks of the random access file. This is illustrated in figure 2.2.

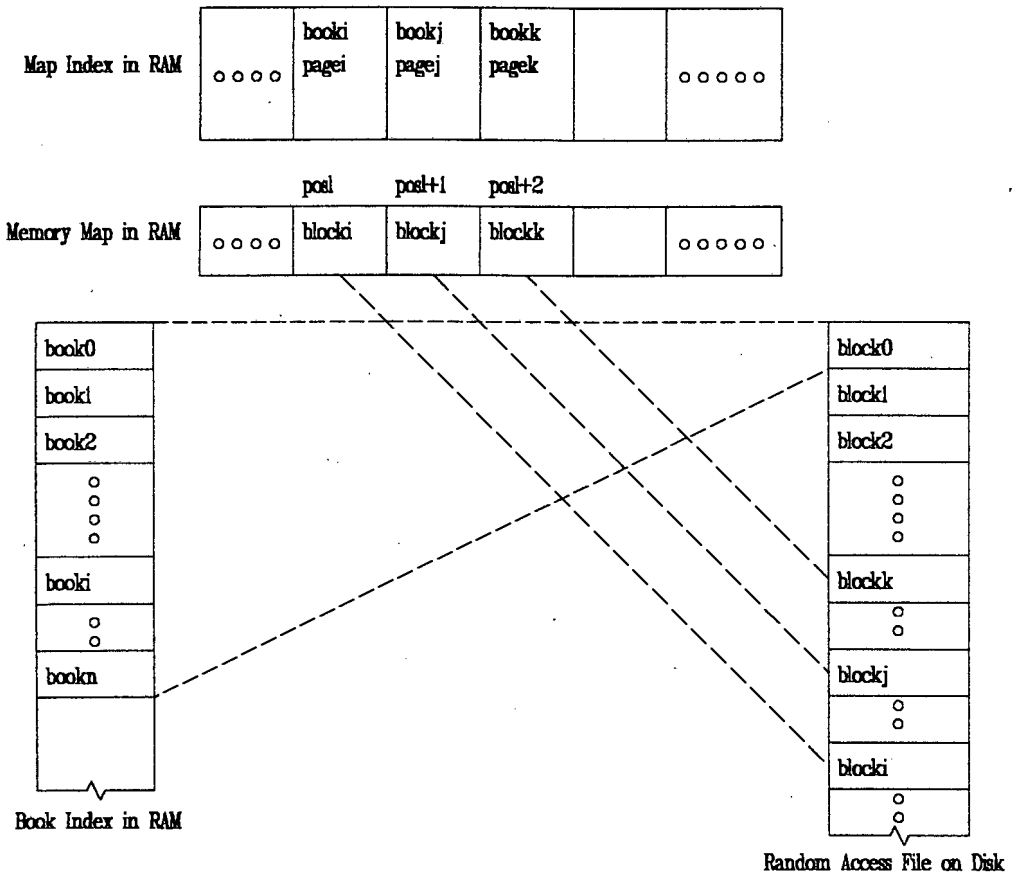


Figure 2.2 : Schematic Outline of Data Handling Scheme.

Consider the memory map in RAM which contains, amongst others, copies of data blocks i, j and k of the random access file in positions l, l+1 and l+2 of the memory map. The map index contains, amongst other information, the book and page number of the data pages as well as the number of the corresponding data blocks currently in each map position. A book index gives access to a pointer system which gives the physical storage address or block number of the random access file for each data page of the data set. The book index is partially contained in RAM and its role is discussed in more detail in the next section.

A data record is referenced by a file-id, a book-id and a record-id. When a record has to be accessed, the system will determine on which page the record resides and search the

memory map to determine whether a copy of that page is currently in the memory map. If this is the case, the record can be accessed directly from the memory map. A **paging function**, based on a **paging algorithm,** will automatically swop pages between RAM and disk if a specific page is not in memory when a record on that page needs to be accessed. This creates a virtual memory for the whole data set or set of random access files. The paging algorithm eliminates unnecessary disk input and output operations. In this context the data handling scheme uses a disk-memory caching technique.

A data record can be referred to in three different ways. The records are numbered sequentially and therefore a data record can be accessed directly by its number. Secondly, a label or key field can be allocated to the data records and therefore a labelled data record can be referenced by this label. A third alternative is for records that contain co-ordinate or geometric fields to be referenced by their geometric position. The data handling scheme further provides for four additional storage schemes, namely stacks, linked lists, references lists and cross referenced lists. These are important for the implementation of sorting and searching procedures.

2.3 The Data Handling Scheme.

The Record Data System is presented in more detail in this section. The basic principles of access to a data record, the concept of the memory map and page swopping are described.

2.3.1 Basic Principles.

The basic workings of the RDS are described below by using the most simple case: the storing and retrieving of a record by its number. This is the basis of the storage scheme and the basic access method.

The page size is of fixed length and is the same for all the books in the RDS. It is most efficient to declare the page size to be a multiple of the block size of the random access files. A data record consists of a fixed number of data fields and therefore has a fixed length. The length of a data record is fixed for a particular book but can vary for different books. The number of records per page will therefore also be fixed for a particular book.

The records and pages of the books are numbered sequentially so that the page number for a particular record can be obtained by a simple calculation. Only those pages that contain data will exist in the random access file and unused storage space will not be allocated when large gaps exist in the sequence of numbers.

The workings of the system are explained with the aid of figures 2.3, 2.4 and 2.5. The elements of the system, as illustrated in figure 2.2, are the **random access file**, the **memory map**, the **map index**, the **book index** and the **book pointers** or pointer tables. Consider the case of record number *recordi* of book number *booki*. The page size is given by *size* and the record length for book number *booki* is given by *leni*. The number of records per page for book number *booki* is calculated as *numi=size/leni* and therefore record number *recordi* will reside on page number *pagei=recordi/numi* where pages and records are numbered from zero. The record number *booki* in the book index contains these and other parameters which constitute a system of pointers to page number *pagei*.

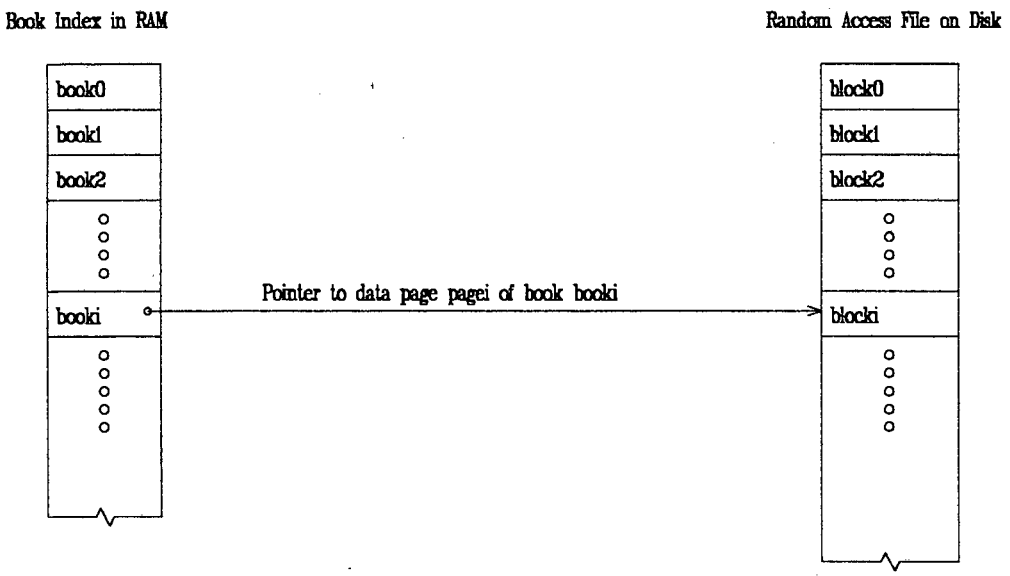


Figure 2.3 : Level 0 Pointer System.

If the required record number *recordi* is less than *numi* and no entry number equal to or greater than *numi* is currently stored in book number *booki*, then the pointer system in the book index entry number *booki* will contain the block number *blocki* of the random access

file which contains page number *pagei* of book number *booki*. In this case one of the parameters in the book index will indicate that the pointer system for this book number *booki* is currently at level zero. See figure 2.3.

If more than *numi* entries are currently stored in book number *booki*, the pointer in the book index entry will contain the block number of a special type of page called a pointer page. The records in this pointer page are pointers with length *pleni* so that the number of pointers per pointer page is given by *pnumi=size/pleni*. If the highest entry number currently stored in book number *booki* is less than *pnumi*numi*, then record number *booki* in the book index will contain the pointer to the pointer page. The record number *precordi=recordi/(pnumi*inum)* in this pointer page will contain the block number *blocki* of the random access file which contains page number *pagei* of book number *booki*. The pointer level parameter in the book entry index will indicate that the pointer system for this book is currently at level one. See figure 2.4

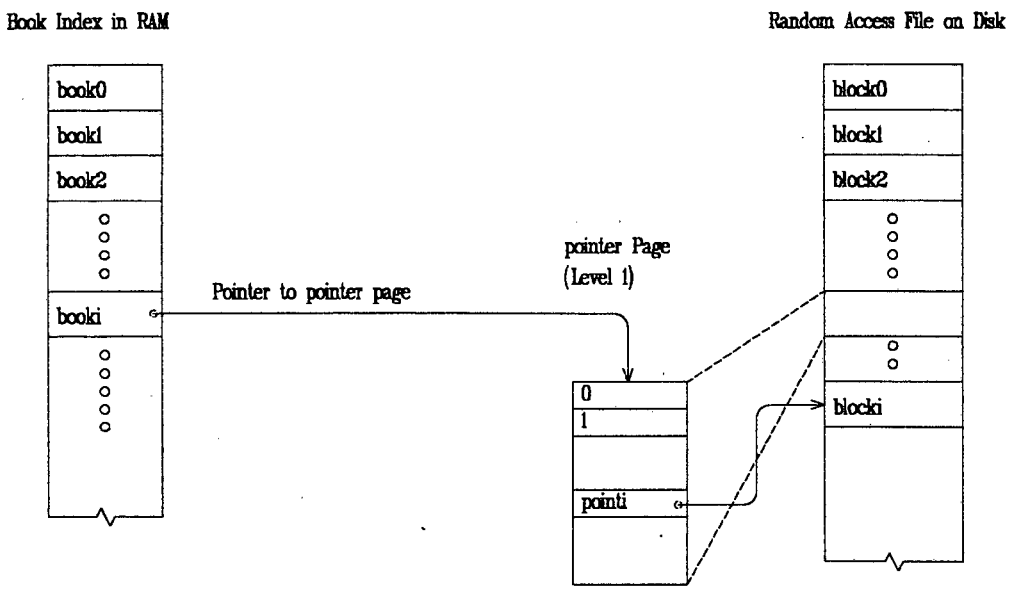


Figure 2.4 : Level 1 Pointer System.

Similarly, if more than *pnumi*numi* entries are currently stored in book number *ibooki*, the pointer in the book index entry will contain the block number of a pointer page at level two. The records in this level two pointer page are also pointers with length *pleni* and with

pnumi=size/pleni pointers per pointer page. The records in this level two pointer page are pointers to the level one pointer pages, which again contain the pointers to the data pages. If the highest entry number currently stored in book number *booki* is less than *pnumi*pnumi*numi*, then record number *precordi=recordi/(pnumi*pnumi*numi)* on the level two pointer page will contain the pointer to the level one pointer page. Record number *precordi=recordi/(pnumi*numi)* on level one pointer page will contain the block number *blocki* of the random access file which contains page number *pagei* of book number *booki*. The pointer level parameter in the book entry index will indicate that the pointer system for this book is currently at level two. See figure 2.5

This concept is extended to develop pointers in pointer pages for as many levels as is necessary to cater for the required entry number in a book. It is important to note that the required number of levels of pointer pages will be generated for a large record number, but

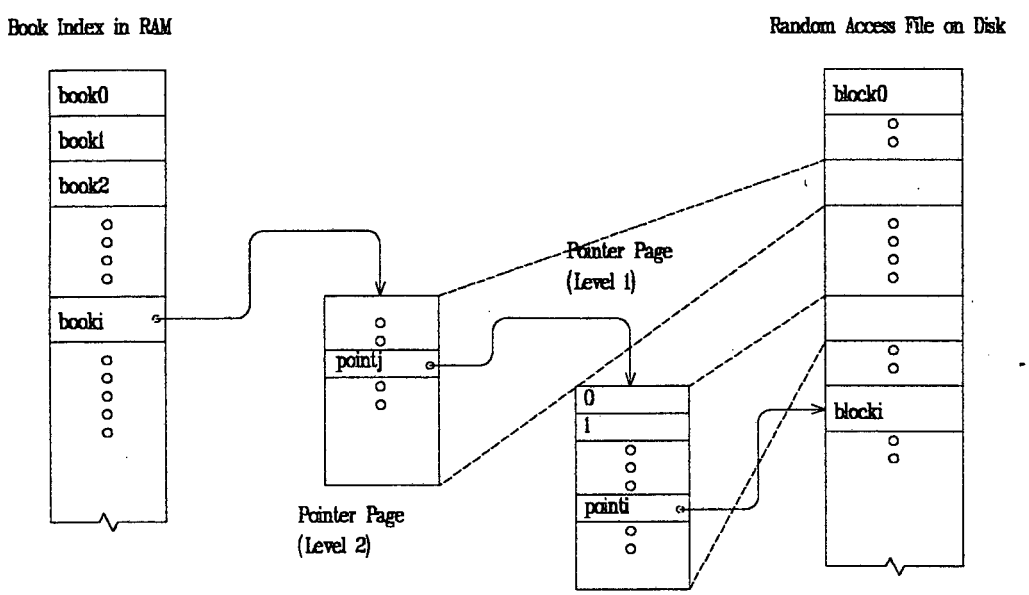


Figure 2.5 : Level 2 Pointer System.

that only those pointer pages and data pages that are actually required will be created in the book. This implies that space will not be allocated in the random access file if it is not actually needed for data pages or pointer pages.

Consider a practical example. The page size is 512 bytes and the pointers in the pointer pages are two bytes long. The data records of a particular book are used to store information concerning the co-ordinate position of ground points. This information is stored in the four following data fields: a co-ordinate pair in two 8-byte words, the ground level in one 4-byte word and a description in a 12-byte character string. This gives a total length of 32 bytes for the data record. This results in 16 records per data page. If fewer than 16 ground points are currently stored and the highest point number is less than or equal to 15, the pointer in the book index will contain the block number of page one of this book, giving access to the 16 data records numbered 0 to 15. If more ground points are stored, or if the highest entry number is equal to or greater than 16, the pointer in the book index will contain the block number for the level one pointer page. The pointers in this level one pointer page can contain the block numbers of the first 256 data pages of this book, giving access to 4096 data entries numbered 0 to 4095. If still more ground points are stored, or the highest entry number is equal to or greater than 4096, the book index pointer will contain the block number of a level two pointer page which, in turn, can contain the block numbers of 256 level-one pointer pages of which each can contain the block numbers of 256 data pages. This gives access to 65536 data pages containing 1048576 data entries.

2.3.2 Page Mapping and Page Swopping.

All the data pages of the data set reside on the random access files at all times. An area in random access memory, called the **memory map**, is reserved to keep copies of, or to map, a number of pages. The memory map, therefore, contains copies of a number of data blocks of the random access file. A table of pointers, called the **map index**, is maintained to keep track of the pages that are currently copied in memory. This is shown schematically in the diagram of Figure 2.2.

Reconsider the case of record number *recordi* of book number *booki*. When this record has to be accessed, the system will determine the block number *blocki* in the random access file which contains page number *pagei* of book number *booki* on which record number *recordi* resides, as described in the previous section. The system will then search through the map index to determine if a copy of this page, i.e. the data block containing this page, is currently in the memory map.

The location or position of record number *recordi* in the memory map will now be explained with the aid of figure 2.6. If page number *pagei* is in memory, it may occupy position *posi* in the memory map where the first memory position is zero. If the length of a record for book number *booki* is *leni*, then the number of records per page for book number *booki* is *numi=size/leni* where *size* is the page size. The record number *recordi* will therefore be in location *loci=mod(recordi,numi)* on page *pagei*. Record number *recordi* will therefore start at the byte following byte *bytei=loci*leni* of page *pagi*. If page *pagi* is in page position *posi* in the memory map, then record number *recordi* will start at the byte following byte number *bytem=posi*pagei+bytei* in the memory map. Record number *recordi* is therefore obtained by retrieving or copying the next *leni* bytes after byte *bytem* in the memory map.

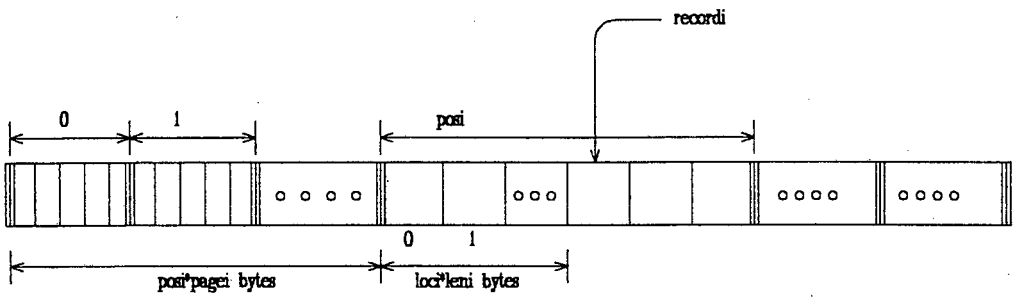


Figure 2.6 : Location of Entry in Memory Map.

If the required page number *pagei* is not in the memory map, the system will copy it to the memory map from the random access file. Space will be created in the memory map for this incoming page by moving one of the pages in the memory map back to the random access file. This is called page swopping and the page in memory that is to be swopped is determined by the paging algorithm. One of the parameters in the map index keeps track of whether any of the entries on a page have been updated during the time that the page resided in memory. Only if an entry on the page to be swopped has been modified will the page actually be copied back to the random access disk, thereby eliminating unnecessary disk access. The page position in the memory map for the page swopping may be given by *posi* and the retrieval of the required record number *recordi* is identical to the procedure described in the previous paragraph.

The paging algorithm is crucial to the efficient operation of the data retrieval system. Various schemes to determine the most suitable page in the memory map for page swopping can be presented. The ideal solution would present the right balance between a simple and computationally fast but probably not very efficient algorithm versus a more complicated and computationally intensive but theoretically optimal solution. The algorithm which is currently employed, and one that seems to give excellent results, is based on a 'least recently used and time in memory' (LRUTIM) scheme. This algorithm, as well as an alternative simple algorithm, was developed by the author and is given in Appendix A.

2.4 Storage Schemes.

The procedure for access to a data record for storage or retrieval, as described in section 2.3, is called access by number. This scheme is based on the fact that each data record has a numerical number or address and the entry position on a page and page position in a book can be calculated from this numerical value. It has also been shown that, although the data records are numbered sequentially, all the numbers in a numerical range need not exist and the system will attempt to allocate storage space only for those data records that actually exist. It will, for example, be acceptable to start numbering a sequence of entries at 2001. The system will, apart from one or two pointer pages, use no more storage space than when the entries were numbered sequentially from 1 because space will not be allocated for data records 0 to 2000. The access by number is the basic storage and retrieval mechanism for data entries. The other access methods are described in the next section.

There are a number of additional storage schemes in the system. These are all based internally on the standard storage scheme of storage by number, but externally offer additional storage schemes. These storage schemes provide powerful tools for the design of flexible data structures.

2.4.1 Stacks

A very simple but quite useful scheme for data storage and retrieval is one in which a book can be declared as a stack. A stack is a list that restricts access to one end. This is called

the top of the stack. The opposite end is called the base. This does not allow any update operations. This can be visualised as a stack where, during storage, data records are always placed on top of the stack. Access is only provided to the top of the stack and during retrieval the top entry on the stack is the one to be retrieved. To access an entry which is lower down the stack, all the entries above it must be removed. This action of removing the top entry on the stack is called popping the stack. The stack may seem quite restrictive but in fact it is a very useful structure. The stack is used in cases where the user is not interested in keeping track of entry numbers and where the entry to be retrieved will always be the last one that was stored.

2.4.2 Linked Lists

Linked lists eliminate the need to store the data records in contiguous locations. It is a simple but very useful storage scheme. In a linked list each data record contains two pointers which indicate the locations of the next and the previous data record in the list. In reality the system will sequentially number and store the data records in the book as they are presented for the first time. This is called the book record number. The linked list allows insertion of a data record at a specific position in the list. This is called the list record number. Insertion or deletion of a data record at a specific list location will update the pointers to maintain the required order.

It is now also possible to sort the list according to a specific rule, for example on the numeric or alphanumeric value of a specific field in the data record. The data records will not be physically sorted but the pointer system of the linked list will be regenerated to reflect the new access order.

The diagram in figure 2.7 illustrates this concept. When a linked list structure is employed in a book, the data records must be declared with a header. This header provides the space required for the pointers of the linked list. The header of each entry contains two pointers. The one pointer points to the next data record in the list while the other points to the previous data record in the list. When adding a data record to the list, it can be placed at the beginning of the list, at the end of the list or at a specific position in the list. Similarly, data

records can be retrieved from the beginning of the list, from the end of the list or from a specific position in the list.

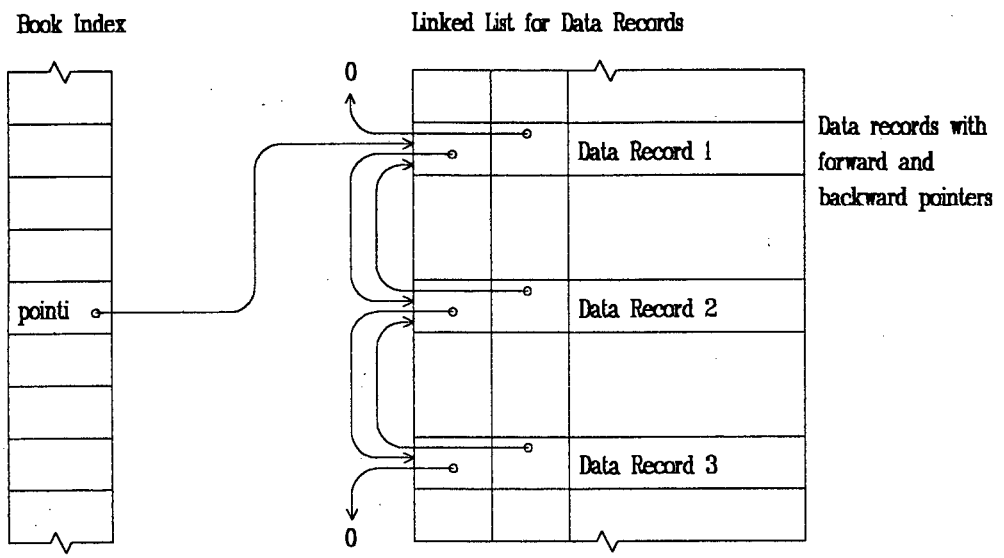


Figure 2.7 : Pointer Scheme for a Linked List.

2.4.3 Referenced Lists.

The Referenced List storage scheme is a slightly more complex scheme than a linked list but offers some powerful capabilities to the user. In the Referenced List storage scheme a number of data records from one book can be attached or referenced to a data record from another book with the aid of a special reference table. This reference table is stored in a separate book. The data records of the book that will be referenced by means of the reference list must be declared with header space to accommodate the pointers. The data records of the reference table book consist of pointers only. This scheme is illustrated in Figure 2.8. Numbers in boxes, diamonds and circles refer to entry numbers in books i, j and k respectively. The entries 1,2,3 and 4 of book j are referenced to entry 1 of book i and entries 3,4,5 and 6 of book j are referenced to entry 2 of book i. The pointers from book i to the reference table in book k will yield a sequence of entries which will give access to the referenced entries of book j.

An example illustrates the practical use of this storage scheme. Consider the situation where information on polygons, i.e. land parcels or lots, are stored in one book and the co-ordinates of the nodes, i.e. the lot corners, are stored in another book. The corners for each lot can be declared in a reference list and this will allow access to the corners of a particular lot via the lot identification in a simple and elegant way. This is achieved through the system of pointers. It is independent of the number of corners per lot and therefore does not require the declaration of arrays dimensioned to cater for the largest possible case.

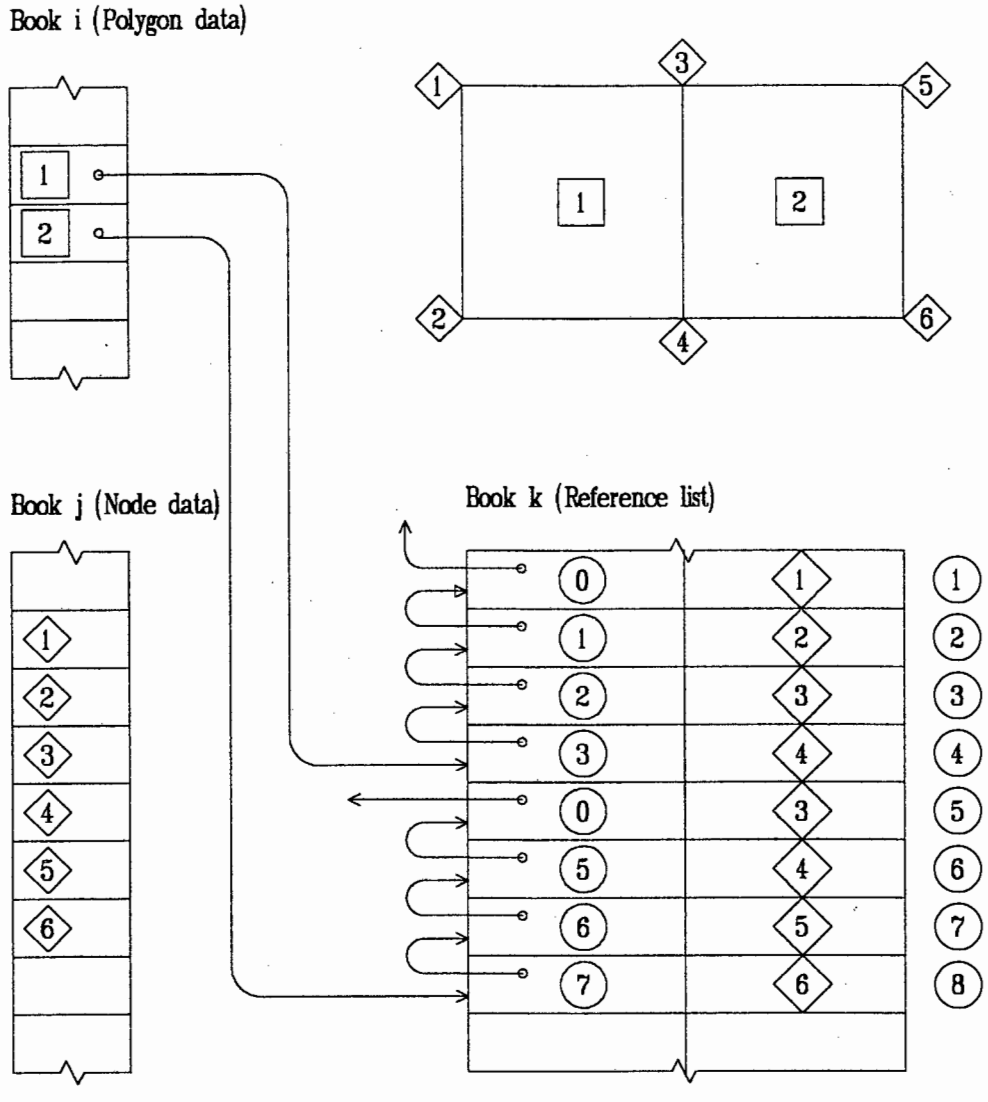


Figure 2.8 : Pointer Scheme for a Reference List.

The pointer in the data record for polygon 1 points to a record in the reference list containing the node number of one of the nodes. This reference list record also contains a pointer to a second record in the reference list containing a second node number of the polygon as well as a pointer to a third record. This process is repeated until all the nodes of the polygon are referenced when the sequence is terminated by a zero pointer. In this way the pointer in polygon record 1 will, via the reference table, yield the node numbers 4,3,2 and 1. Similarly the pointer in polygon record 2 will yield the node numbers 6,5,4 and 3

2.4.4 Cross Referenced Lists.

The Cross Referenced List storage scheme is a more complex extension of the Referenced List storage scheme but offers very powerful capabilities to the user. In the Cross Referenced List storage scheme a number of data records from one book can also be attached to a data record from another book with the aid of a special cross reference table. This cross reference table is also stored in a separate book and has a more complex structure than the standard reference table. The data records of both the books that are cross referenced must be declared with header space to accommodate the pointers. The records of the cross reference table book consist of pointers only.

This scheme is illustrated in Figure 2.9. Numbers in boxes, diamonds and circles refer to data record numbers in books i, j and k respectively. Data records 1,2,3 and 4 of book j are referenced to data record 1 of book i and data records 3,4,5 and 6 of book j are referenced to data record 2 of book i. The pointers from book i to the reference table in book k will yield a sequence of records which give access to the cross referenced data records of book j. These are called the primary references. Similarly, the pointers from book j to the reference table in book k yield a sequence of records which give access to the cross-referenced data records of book i. These are the secondary references.

An example illustrates the practical use of this storage scheme. Consider again the case where information on polygons representing land parcels or lots is stored in one book and the co-ordinates of the nodes, i.e. the lot corners, are stored in another book. The lot corners for each lot can be declared in a cross reference list and this allows access to the lot

corners of a particular lot via the lot identifier (erf number). The cross referenced list also allows an additional facility in the inverse form. It is now also possible to access all the land parcels or lots that have a particular common corner via the lot corner identification. This is all done through the system of pointers which is independent of the number of corners per lot. As with referenced lists, it does not require the declaration of arrays dimensioned to cater for the largest possible case.

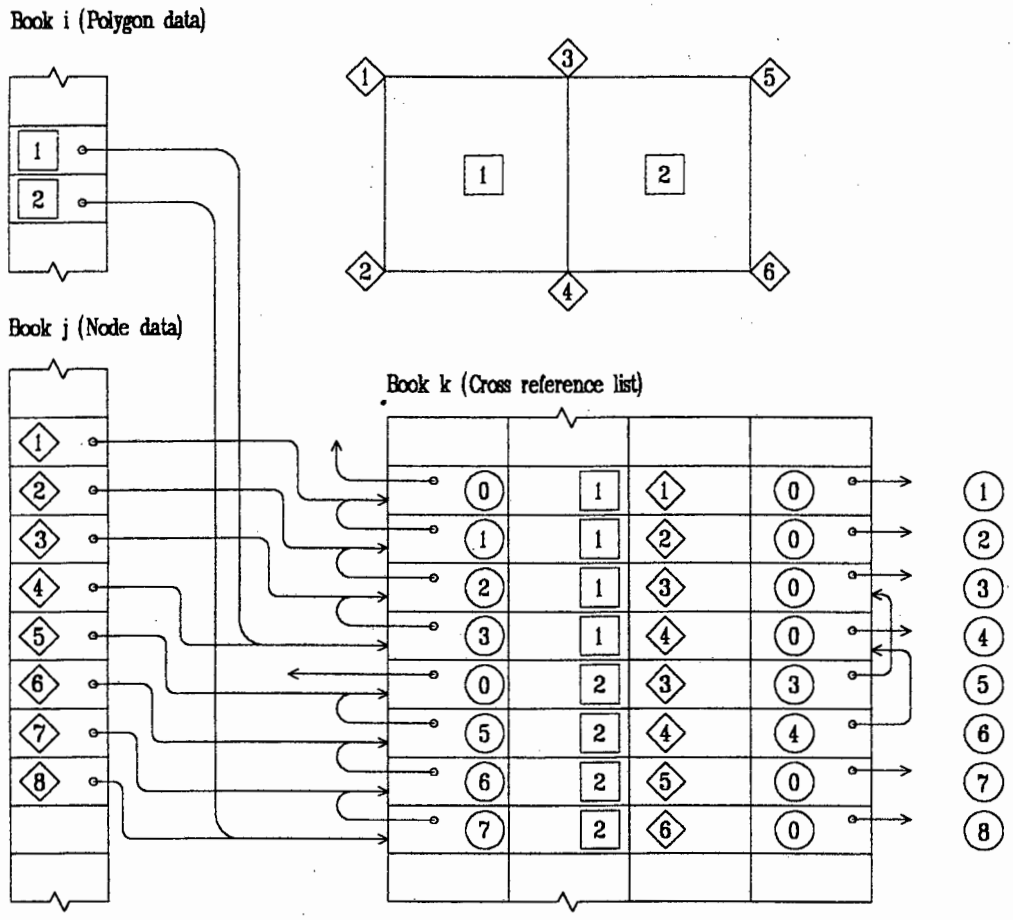


Figure 2.9 : Pointer Scheme for a Cross Referenced List.

In the primary references, the pointer in the data record for polygon 1 points to a record in the cross reference list containing the node number of one of the nodes. This cross reference list record also contains a pointer to a second record in the cross reference list containing a second node number of the polygon as well as a pointer to a third record. This process is repeated until all the nodes of the polygon are referenced when the sequence is

terminated by a zero pointer. In this way the pointer in polygon record 1 will, via the primary references of the cross reference table, yield the node numbers 4,3,2 and 1. Similarly the pointer in polygon record 2 will yield the node numbers 6,5,4 and 3

In the secondary references the pointer in the data record for node 3 points to a record in the cross reference list containing the polygon number of a polygon containing this node. This reference list record may also contains a pointer to a second record in the cross reference list containing a number of another polygon common to this node as well as a pointer to a third record. This process is repeated until all the polygons common to these nodes are referenced. Then the sequence is terminated by a zero pointer. In this way the pointer in node record 3 will, via the cross reference table, yield the polygon number 2 and polygon number 1. Similarly the pointer in node record 4 will also yield the polygon number 2 and polygon number 1 while node 5 will yield only polygon number 2 and so on.

2.5 Access Methods.

The storage scheme of storage by number is the basic storage method and, consequently, the basic access method. Two alternative methods of access are provided in the RDS. In the first alternative a character string or label can be used as a key field so that an entry can be stored and retrieved by this key. In the second alternative, a data record containing a geometric position can be stored and retrieved by means its geometric position.

2.5.1 Access by Name.

It is often desirable to refer to a data record by a name or label. It is inefficient to search through a long list of records when a record with a particular name or label has to be found. A more direct method of record access via a label is required. The RDS provides a facility for identifying a data record with a name or label and then storing or retrieving the record by that label. The method used is a modified hashing technique.

In a key-to-address transformation or hashing technique, a formula is applied to the label field in order to generate a number in a pre-defined range. The simplest solution would be to store the record by this number but with the obvious limitation that more than one label

or key-field can yield the same hash number. This is called a collision. Since the purpose of hashing is to squeeze a potentially large key space into a small address space, it is inevitable that collisions will occur. That is, two or more keys will transform into the same address or record number.

Various techniques can be applied to eliminate this problem. One popular alternative is to provide storage for more than one data record at each hash number. This is called a hash bucket. The theory is that only a limited number of entries will end up in each hash bucket and therefore a limited number of searches will be required to locate the target entry. This is essentially the same as enlarging the range of possible hash addresses. A second alternative is, in the case of a collision, to check the next and subsequent addresses, until a vacant spot is found. A variation on this theme, called pseudorandom probing, is to compute a new offset location to be added to the hash address until a vacant spot is found, instead of using the next address. The drawback of these schemes is having to calculate more and more offsets as the storage range fills up. A further shortcoming is that a limited finite storage range, which must be quite large to eliminate these potential problems, must be defined.

An effective solution to these problems is to use the hashing technique in combination with a linked list. The hashing technique is used to generate a numerical address in the limited address range, also called the hash table. The entry in the hash table now contains a pointer to the first data record of a linked list. This results in the reservation of a moderate workspace for the hash table and an open ended linked list instead of fixed size hash buckets.

This scheme is illustrated in figure 2.10. This requires the use of two additional books for the hash buckets and linked list pointers respectively.

A very important aspect is a suitable formula or algorithm for the hashing function. If several keys hash to the same address, the linked list becomes lengthy and the sequential searching which follows becomes too costly. This occurs when the hashing function does not spread the addresses uniformly throughout the address range. This effect is called clustering.

The labels or keys are limited to eight characters in the RDS and the ASCII values of the characters are used to calculate the hash number. Two examples of hashing functions are detailed in Appendix B.

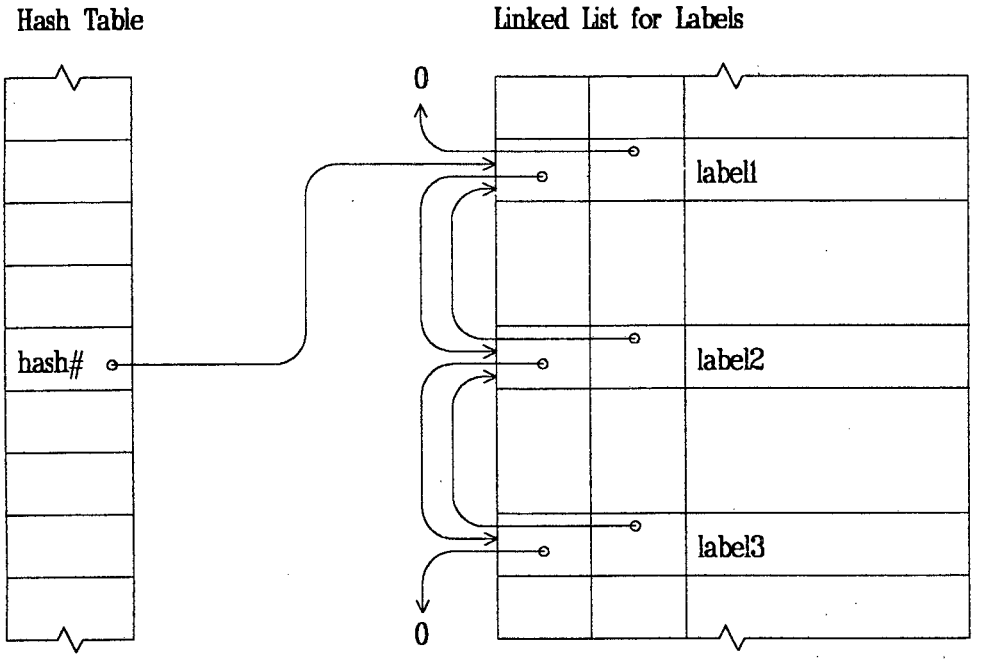


Figure 2.10 : Hash Table with Linked List.

2.5.2 Access by Position.

When working with geometric information, it is often necessary to search for a point or geometric object in a specific position or in a certain region. The problem again arises that it is inefficient to search through a large list of data records each time this is required. The RDS provides a facility for providing access to a data record via the geometric position data when a co-ordinate pair is part of the data record. Two schemes are employed to provide this facility.

The first scheme used for data access by geometric position follows arguments similar to those used in the case of access to a data record by a label field. The key-to-address transformation or hashing formula is replaced by a cell technique. The geometric region is

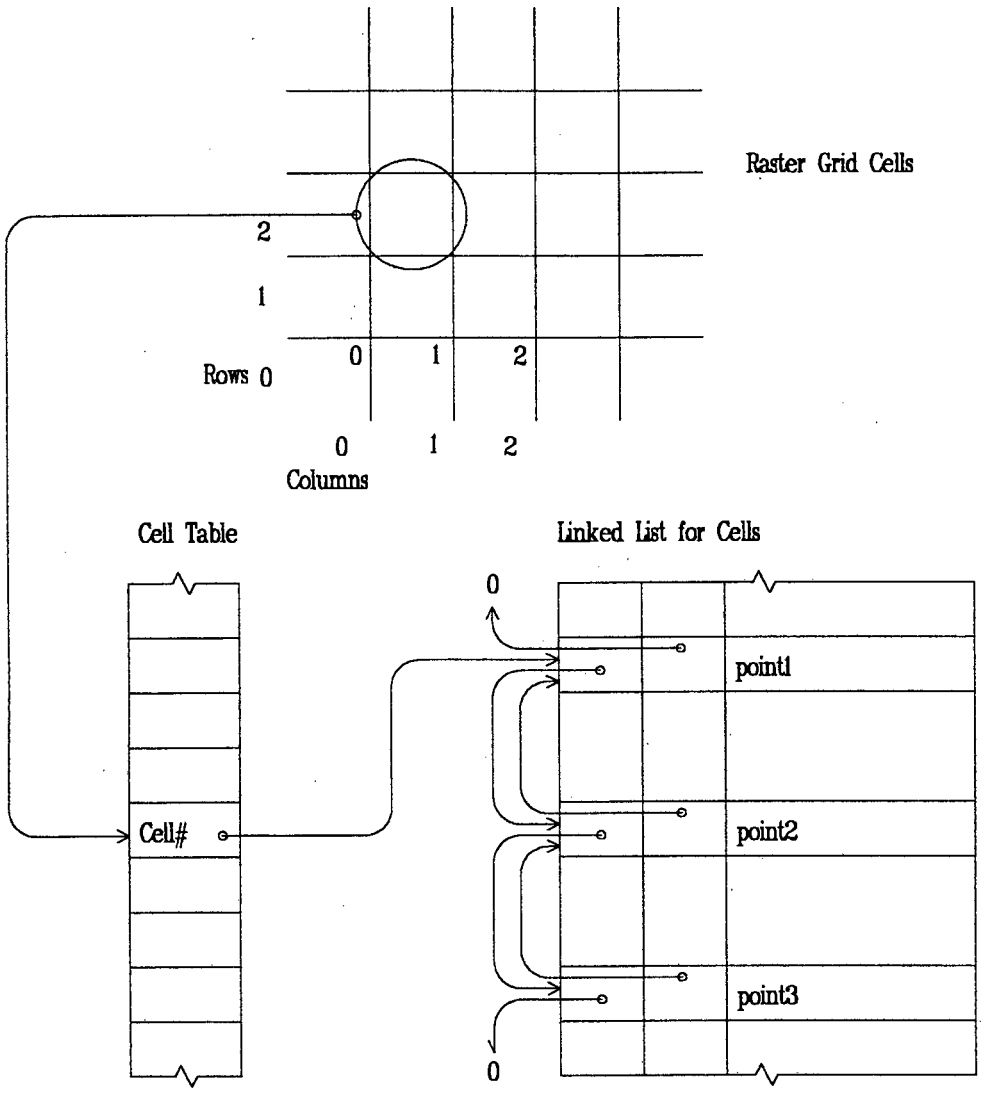


Figure 2.11 : Cell Table and Linked List.

overlaid by a square raster grid and all the grid cells are numbered sequentially, giving an address number in the pre-defined address range, called the cell table. This entry in the cell table now contains a pointer to the first data record of a linked list. The system returns the cell number for all points falling in a grid cell and a limited search will give access to the

required data record. This results in the reservation of a moderate workspace for the cell table and an open ended linked list for all the points in a grid cell. This scheme is illustrated in figure 2.11. This requires the use of two additional books for the cell table and linked list pointers respectively. This scheme works very effectively in most of the situations where the points are evenly distributed over the geometric region. A greater concentration of points in some regions and a sparseness of points in other regions can lead to clustering which in turn leads to long pointer lists. An added shortcoming is that prior knowledge of the range of the co-ordinate values is required for the definition of the raster grid. This is overcome to a certain extent by the use of open edge cells, as shown in figure 2.11.

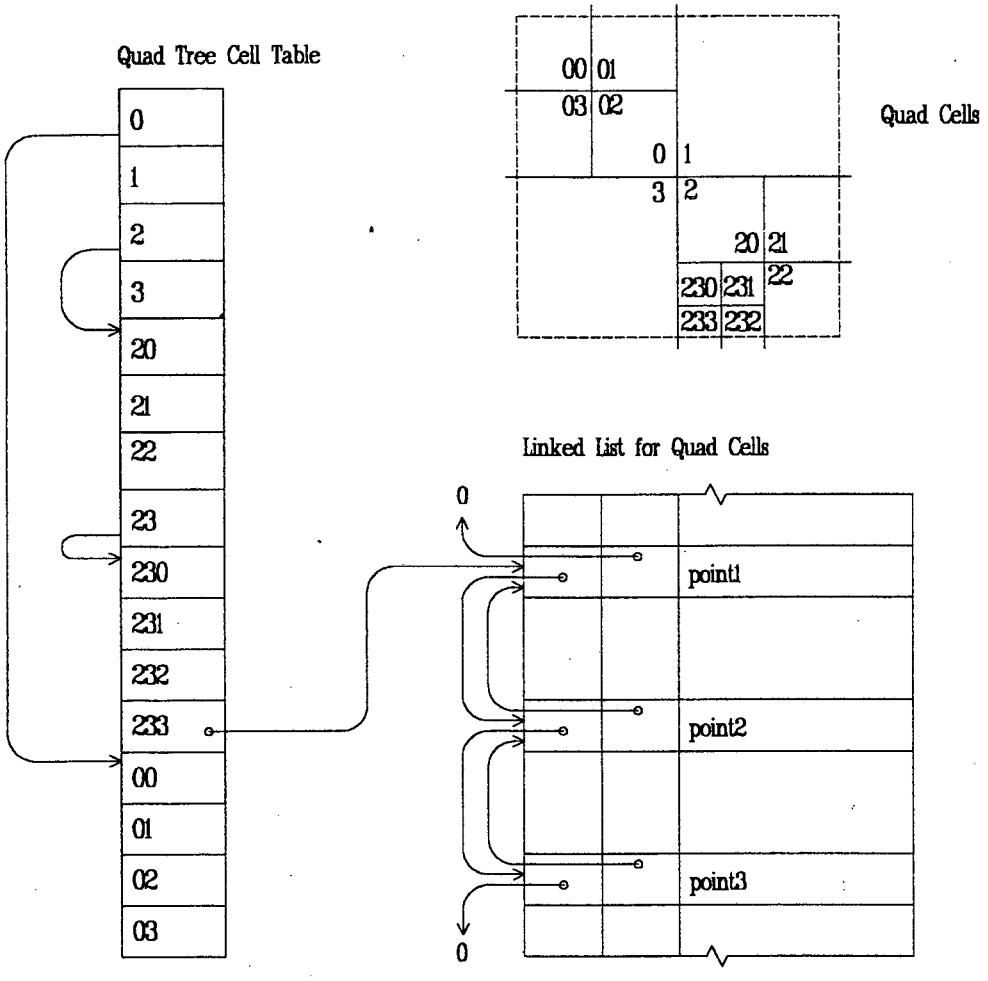


Figure 2.12 : Quad Tree Cells with Linked List.

A second scheme for data access by geometric position is a variation of the first one, and strives to overcome the above mentioned shortcomings. This scheme uses a variation of the quad tree instead of the grid cells. The fixed-length cell table is replaced by an open ended quad table. The region is initially divided into four open quadrants or quad cells, and the position of the origin is not critical. The four quad cells correspond to the first four entries in the quad table. These entries contain the pointers to the first data record in the linked lists for each quad respectively. When the number of points in a quad exceeds a pre-defined threshold, i.e. the length of the linked lists exceeds a pre-defined limit, the quad is subdivided into four quadrants. These new quad cells now each contain a sub-set of the previous set of points. The pointer in the quad table is changed to point to a new quartet of entries and these new entries contain pointers to the first entries of the new linked lists corresponding to each subset of points. This scheme is illustrated in figure 2.12.

2.6 Procedures.

The data handling and data storage schemes described above are implemented in computer programs through a set of computer procedures or subroutines. The procedures have been assembled in a library of procedures and can easily be incorporated in C, FORTRAN or Pascal programs. The procedures have been developed, tried and tested over a number of years. They have proved to be an invaluable aid in the development of the algorithms and programs described in the following chapters.

The following list gives the name and purpose of the most important procedures or subroutines in the RDS system and gives an indication of their usage.

inirds	Initialise (create) a data set
addrds	Add a file to the data set
stords	Store the data set (force flush)
clords	Close the data set
putnum	Store a data record by number
getnum	Retrieve a data record by number

putlab	Store a data record by label
getlab	Retrieve a data record by label
putpos	Store a data record by position
getpos	Retrieve a data record by position
putstk	Place a data record on a stack
popstk	Pop a data record from the stack
putlnk	Add a data record to a linked list
nxtlnk	Retrieve the next data record in the linked list
srtlnk	Sort a linked list
putlst	Add a reference in a reference table
getlst	Retrieve a reference table
putref	Add a cross reference in a cross reference table
getpri	Retrieve the primary references from a cross reference table
getsec	Retrieve the secondary references from the cross-reference table
delnum	Delete a data record stored by number
dellab	Delete a data record stored by label
dellst	Remove a data record from a linked list
dellnk	Delete a reference in a reference table
delref	Delete a cross reference in a cross reference table

CHAPTER 3

The Geometric Data Structure.

3.1 Introduction.

The description of the real world in a GIS is called an application model or a geometric model. Such a model denotes descriptions of objects and relationships relevant to a given application and is stored as a structured collection of data called the model data structure. These data are stored in the two distinct data bases of the GIS. The geometric data base and the alpha-numeric data base.

Geometric models describe entities with inherent geometric properties and naturally lend themselves to graphical representation. The physical features of the real world are represented in a GIS by graphic entities in a geometric data base. Typical examples of some of these features are land parcels, sewer or water pipes, electrical cables, road signs, water meters, valves and fire hydrants. The basic graphic entities are polygons, lines and points. The graphic entities are connected to the descriptive information about the features in the alpha-numeric data base. Typical examples of these are information on pipe, cable and construction material, equipment, size, type, serial number, date serviced and other management information that needs to be maintained in the data base.

3.2 The Base Map.

A distinction is drawn between two sets of graphic entities. These are called the primary graphic entities and secondary graphic entities. The primary graphic entities form the geometric reference framework on which the application area under consideration is modeled. This is called the base map and consists of polygons that do not overlap, lines that do not intersect each other and lie on the edges of polygons and points which are at the apexes of polygon boundaries or ends of a line.

The secondary graphic entities model the remaining geometric features of the application model relative to the base map. The secondary entities are polygons that may overlap other polygons, lines that may cross other lines or traverse polygons and

points that may fall inside a polygon or on a line. The secondary entities, therefore, overlap the primary entities or base map and can also overlap other secondary entities.

The cadastral layout constitutes the base map in a municipal GIS. The base map will consists of one or more outside figures which contain the blocks and road reserves. Excluded areas are treated as an additional outside figure which do not contain any blocks and road reserves. The blocks are divided into the individual lots or land parcels. The lots fill the blocks completely and do not overlap nor may there be a gap between two parcels with a common boundary. The blocks together with the street reserves fill the outside figure completely and also do not overlap. The points, lines and polygons which define the base map are stored with full topology. Therefore, recalling a polygon will provide all its points and lines. Recalling a point will provide all the polygons or lines common to that point. Recalling a line will provide the polygons adjoining that line. The lines and points can be classified according to the polygons to which they belong as outside figure edges and points, block boundaries and block corners and lot boundaries and lot corners.

The remaining geometrical entities are represented by the secondary graphic entities. For example, servitudes, lease areas or the outlines of buildings can be represented by secondary polygons that overlap the land parcels. Pipes and cables of the water, sewer or electrical reticulation networks can be represented by secondary lines that can traverse the land parcels and intersect block or lot boundaries or each other. The normal topological information is also stored for these entities. Therefore, recalling an entity will provide information on connected, adjoining or related entities. The secondary entities are also classified according to the polygons to which they belong. For example, polygons could be servitudes or buildings, while lines could be servitude boundaries, building edges, pipes or cables and points could be valves, manholes, traffic signs, lamp posts or water meters.

3.3 Data Structures.

The application model in a GIS contains a description of both the graphical and non-graphical properties of objects. The object descriptions are stored in a structured collection of coordinated data called the model data structure. A geometric model is constructed in a geometric data structure. The geometric model is implicitly equated with the geometric data structure of the geometric data base. Data capture for a GIS

implies the compilation of all the information and relationships that are required to construct the geometric model. The data capture system, therefore, also needs a geometric data structure or data base.

The data handling scheme, as described in the previous chapter, provides a powerful set of tools to construct a flexible and powerful data structure for the local authority GIS.

The graphic primitives which represent the physical features in the data base are points, lines and polygons. These primitives are numbered sequentially, and can be stored and retrieved by this number. They can also be assigned a key field and can thus be referenced by a name or label. This can be accomplished through the standard storage and retrieval schemes of the RDS.

Polygon entities and line entities, by means of their vertices or end points, and point entities can be referenced via their geometric positions. This, together with the cross referenced storage scheme, allows for an efficient and effective method of defining, storing and retrieving the topology. This also provides the basis for efficient sorting and searching techniques. An extension of the sorting and searching schemes, together with certain raster algorithms, is used in implementing efficient polygon overlay and analysis algorithms.

Other aspects that may require consideration in the design of the data structure are clipping and windowing algorithms, geometric transformations, vector and raster algorithms and display and transformation algorithms.

The sources of information are usually varied in origin and age and therefore often contain different units and map projections or reference systems. These factors will influence the data structure if provision is made to store data in their original units and reference systems. The data structure should also provide for the need to transform the data to a chosen set of units and reference system.

3.4 Data Structure for the Cadastral Base Map.

The cadastral layout forms the base map of a municipal GIS. The cadastral layout is ideally suited to a hierarchical data structure. A new data set is used for each manageable portion of the town or data capture project. This is usually a township or

municipal area represented on a General Plan. The hierarchical data structure is shown in Figure 3.1.

The top of the hierarchy, or root of the structure, is the project or area under consideration. This area is usually a township. The township is bounded by one or more outside figures or outside boundaries. Each outside figure is represented in the data structure by a polygon. This polygon is defined by the co-ordinated boundary points and bounded by the outside figure boundary lines joining the boundary points.

The outside figure contains the street blocks and road reserves. These areas are also represented in the data structure by polygons which are defined by the block corners and bounded by the block boundaries which are represented by lines joining the block corners.

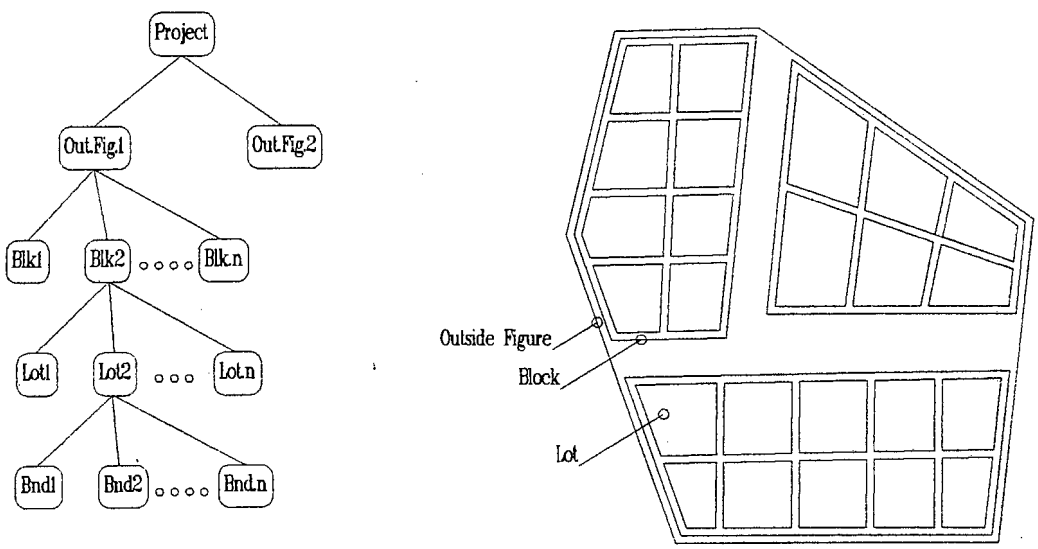


Figure 3.1 Cadastral Hierarchy.

The blocks contain the land parcels or lots. The lots are represented by polygons and defined by the lot corners and bounded by the lot boundary lines or sides joining the lot corners. Original land parcels or lots can be subdivided into one or more subdivisions. The portion of the original lot that is left over is called the remainder. Depending on how the subdivisions are set out, the remainder can consist of more than one portion. The original land parcels, subdivisions and remainders are called the lots. All of these are treated on the same level in the data hierarchy and are represented by polygons.

These polygons are defined by the lot corners and are bounded by the sides of the lots. The lot boundaries are classified as original lot or subdivision boundaries.

The rest of this section gives a more detailed description of the data structure for the cadastral data in the data capture system. This will illustrate the data structure and how the data handling scheme is used.

The first three books are used to store data concerning the co-ordinates of points on the ground. All the co-ordinates are stored in book 1, and a parameter in the data records is used to identify the co-ordinates as outside figure points, block corners, lot corners etc.. Co-ordinated points are identified by an alpha-numeric label and are stored by name. This requires three books. The co-ordinates are stored on book 1, the hash table is stored in book 2 and the reference table in book 3.

The township can consist of more than one outside figure. The outside figure polygons are numbered sequentially, and are stored by number in book 6. A reference table is defined to relate each outside figure polygon to its outside figure co-ordinated points, which are stored in book 1. Each outside figure can consist of a number of street blocks and road reserves. The block and road reserve polygons are numbered sequentially, and are stored by number in book 7. A reference table is defined to relate each block or road reserve polygon to its block corner points, which are also stored in book 1. Another reference table is defined to relate each outside figure to its blocks and road reserves. Similarly, each block consists of the lots. The polygons for the lots are stored in book 8, and a reference table is defined to relate each block to its lots. The data records for the lots are stored by name and therefore need two additional books, one for the hash table and one for the reference list. The data for the lot boundaries are stored in book 9, and a reference table is defined to relate each lot to its sides or boundaries.

This data structure makes it possible to access the required data either directly or through the hierarchy. It is possible to extract the data for the lots or points directly by name. Furthermore, it is possible to get a list all the boundary points of an outside figure as well as all the blocks encapsulated within an outside figure. It is also possible to extract a list of all a block's corner points, as well as all the lots belonging to a block. It is then possible to extract all the sides belonging to a lot. The point name of the starting point of each side is stored as one of the data items in this data record, and provides access to all the points of a lot.

The data structure described above provides for the input data. Additional calculations are performed in the data capture process.

Provision is made in the data structure for the sides of the lots to be stored as the joins, i.e. the distances and bearings, between lot corners. The original lot boundaries are referred to as lot joins or dimensions. It is quite possible that two adjacent lots can have different legal lengths for the same common boundary as depicted on the diagrams framed after independent surveys. The data structure provides for the storage of the full original document data to cope with this ambiguity.

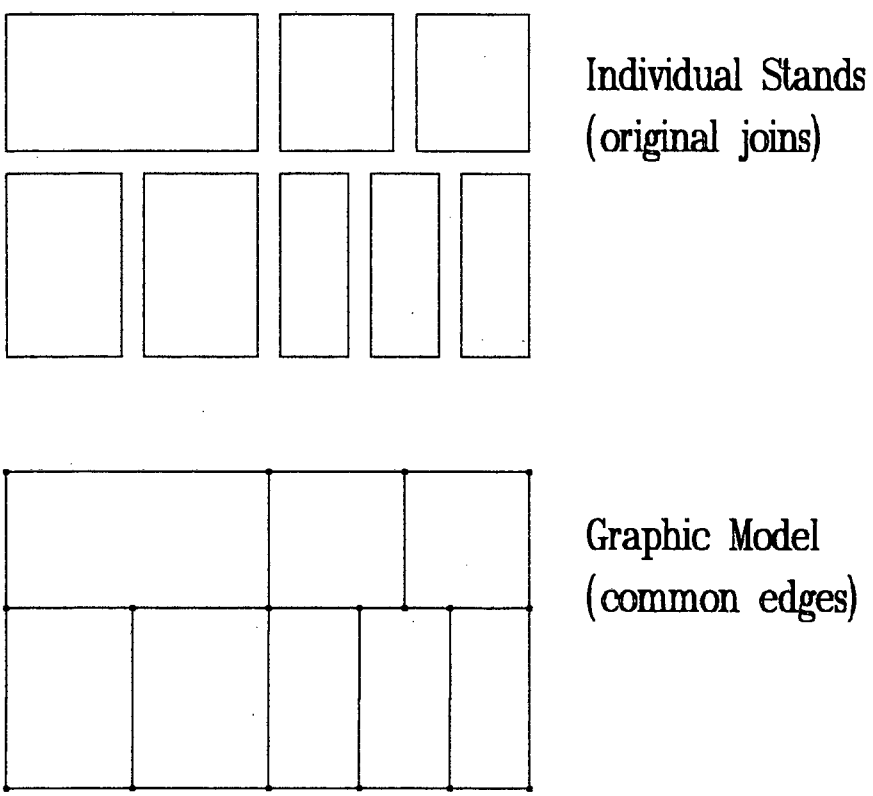


Figure 3.2 Individual Lots and Line Segments.

A further complication arises where more than one lot on one side shares a boundary with a lot on the other side. This is best explained by the example in figure 3.2. The full topology of the cadastral layout requires knowledge of all lots adjacent to all lot boundaries. The data structure should provide for this. The graphic model can only have a single line as a common border between two polygons. New graphic entities

are defined. These are called the graphic points and graphic boundaries. These graphic points are the averaged co-ordinates and the graphic boundaries are the averaged boundary line lengths. They are further broken into line segments at each lot corner point so that a boundary line can have at most only two common lots, namely one on each side. The data structure also provides for the storage of these boundary segments, and a cross reference table is created to define the relation between the lots and the segments. This allows for the extraction of all the segments forming the boundary of a lot. The inverse of this is obtaining the two lots on either side of a boundary segment. Cross reference tables are also defined for the relations between the graphic points and the lots, and the graphic points and the segments. This allows for the extraction of all the corner points of a lot or all the lots common to a corner point as well as the end points of a boundary segment, as well as all the boundary segments common to a point. These properties of the data structure greatly enhance the searching and sorting procedures when performing various calculations, which are described in the next chapter.

3.5 Data Structures for Engineering Services.

The data structures for the various engineering services and reticulation networks are similar to those of the cadastral information. The engineering services also use a hierarchical data which is depicted in figure 3.3. The data structures for the various services are identical, and a sewer reticulation network will be used as an example.

A sewer reticulation network has a tree structure. Some sewer network terminology needs to be declared. The two main components are the nodes or manholes and the edges or pipes. The section between two manholes is called a pipe segment or a pipe for short. A chain of pipes is called a line. Manholes are placed at all points where the line changes direction or where a pipe segment becomes too long. Lines may only join up with other lines at a manhole. The data for the sewer network consist of various items of information about each manhole and each pipe segment.

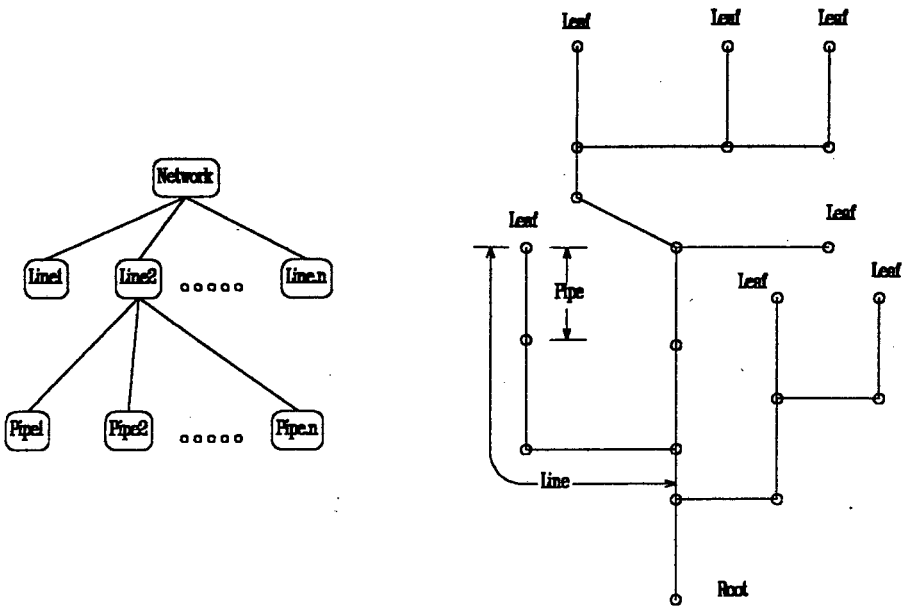


Figure 3.3 Sewer Network Hierarchy.

The hierarchical data structure is set up in a number of data books and reference tables. Certain data for the network as a whole are required and can be stored in, for example, book 65. The network consists of a number of lines and a data entry for each line is stored in book 67, while a reference table defines the relation between the network and its lines. The lines consist of the pipe segments and the data for each pipe segment are stored in book 68, while a reference table is used to give the relation between the pipe segments and the lines. The pipe segments are defined by the two manholes at the endpoints of each segment. The data for the manholes are stored in book 69, while a cross reference table stores the relation between the pipe segments and the manholes or nodes. This hierarchical scheme of data books and reference tables gives access to all the lines of a network or tree and all the pipe segments of a line. The cross reference tables give access to the manholes of a pipe segment or inversely to the pipes joining up at a manhole.

The data structure for the other engineering services follows the same pattern. The storm water drainage system, like the sewer drainage network, is a tree structure, while the road, water and electrical reticulation networks have closed loop patterns. For data storage purposes these closed loops are of a hierarchical structure where the network

consists of pipe, cable or street lines; pipe, cable or street segments and nodes at street intersections, pipe junctions, pipe bends, valves, terminals, cable junctions, transformers etc.

3.6 A Data Structure for Polygon Algorithms.

It is often necessary to find the intersection of two lines or the polygon traversed by a line or to determine if two polygons overlap. It would be time consuming to search through all the possible lines and/or polygons. A data structure which is analogous to the access by position storage scheme for points, greatly reduces the need for this. This powerful data structure, which can be applied in a number of polygon algorithms, is a combination of the referenced storage schemes and raster algorithms.

3.6.1 The Grid Cell Reference Scheme for a Line.

This reference scheme is based on a cell technique. The geometric region is overlaid by a square raster grid. Each grid cell is given a unique number which can easily be calculated from the row and column position. The cell number l at the intersection of row i and column j is given by l=i*n+j where n is the number of rows and columns and i and j can occupy the range 0 to n-1. All the grid cells that are traversed or partially traversed by the line are determined and a cross reference table which defines the relations between the grid cells and a line, is built. This is shown in figure 3.4.

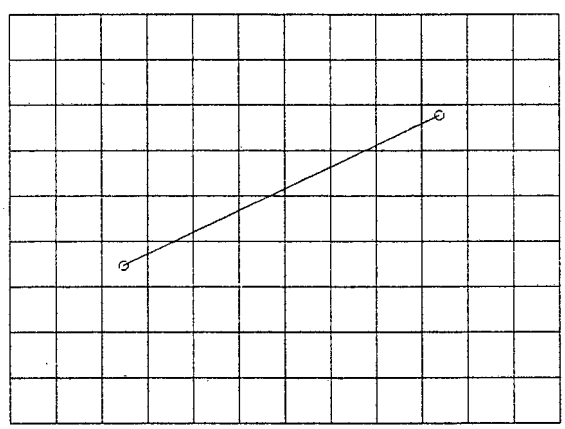


Figure 3.4 Line - Grid Cell Overlay.

It is now possible to extract all the cells that are traversed by a line. It is similarly possible to find all the lines that traverse any particular cell. This data structure can be used to good effect in a number of ways. One example is in using a digitiser to indicate a line. The cell number of the digitised point is determined and all lines traversing this cell can be retrieved in order to calculate the closest line. This allows for a great reduction in the number of lines to be searched as opposed to a search through all the lines. A second example is the requirement to determine all lines that intersect a particular line. A first step is to obtain all the cells which are traversed by the line in question and then to determine all the other lines that traverse each cell and to test for intersections between the lines. This again allows for a great reduction in the number of lines to be searched for the intersection of two lines.

The algorithm whereby it is determined which grid cells are traversed by a line, is based on Bresenham's algorithm for generating the pixels which represent a straight line on a raster display device. (Foley and Van Dam, 1983, and Newman and Sproull, 1979). A more detailed description of this algorithm is given in Appendix C.

3.6.2 The Grid Cell Reference Scheme for a Polygon.

This reference scheme is also based on a cell technique. The geometric region is overlaid by a square raster grid. Each grid cell is given a unique number which can easily be calculated from the row and column position. The cell number l at the intersection of row i and column j is given by l=i*n+j where n is the number of rows and columns and i and j can occupy the range 0 to n-1. All the grid cells that are overlaid or partially overlaid by the polygon are determined and a cross reference table, that defines the relations between the grid cells and the polygon, is built. This is shown in figure 3.5.

It is therefore possible to extract all the cells that are enclosed or partially enclosed in the polygon. Similarly, it is also possible to determine all the polygons that overlap or partially overlap any particular grid cell. This data structure can be used to good effect for a number of searches. It can, for example, be used when the overlapping of polygons or the traversing of polygons by lines has to be determined. The number of polygons or lines to be searched and tested for overlapping or traversing can greatly be reduced. This is accomplished by first determining all the grid cells that are overlaid or

partially overlaid by the polygon, or traversed and partially traversed by the line. It is then possible to obtain all other polygons or lines referenced to each grid cell and test for the required overlap or traversing.

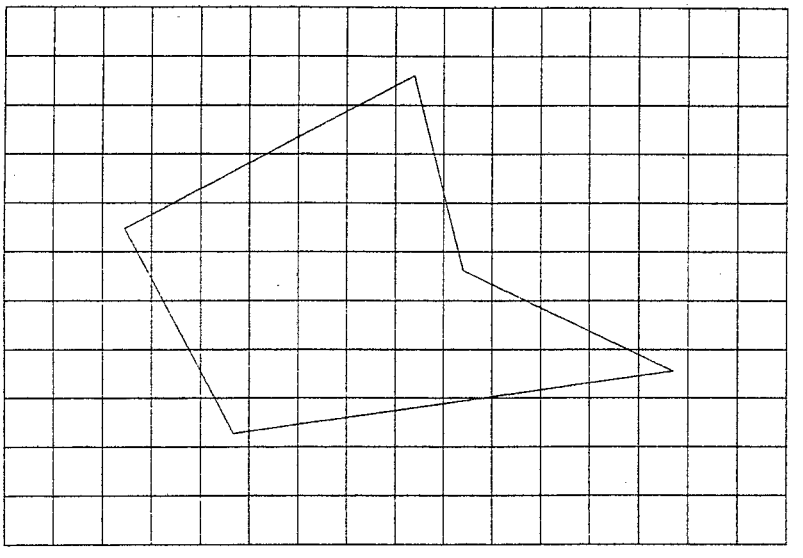


Figure 3.5 Polygon - Grid Cell Overlay.

The algorithm, by means of which it is determined which cells are overlaid by a polygon, is based on a combination of Bresenham's algorithm for generating the pixels representing a straight line and a scan-conversion algorithm to fill a polygon on a raster display device. . (Foley and Van Dam, 1983, and Newman and Sproull, 1979). A more detailed description of the algorithm is given in Appendix D.

CHAPTER 4.

Data Capture of Cadastral Data.

4.1 Introduction.

Data capture of the cadastre for the base map of a GIS is a very important part of the initial data capture effort. Procedures are described which facilitate the efficient and accurate capture of this data. The preferred sources of cadastral data are general plans and survey diagrams but other maps or drawings may also prove to be suitable sources of data.

4.2 Data Capture from General Plans and Diagrams.

The most accurate sources of data for the base map of a local authority GIS are the general plans and survey diagrams which are filed at the offices of the Surveyors General. A methodology has been developed for the capture of data from these source documents in an efficient and cost effective way. The procedures are performed in three phases using a suite of three computer programs. These are referred to as the Data Typing , the Geometric Calculation and the Graphics Calculation phases respectively. The computer programs for each phase are designed for use by operators with fair typing skills but low numeracy skills. This means that the bulk of the data capture work can be delegated to lesser trained or experienced personnel and that higher skilled persons are only required at a limited number of data processing and quality control stages in the process. This can equate to major cost savings over the life of a project.

In the first phase, a special data typing program is used to gather the data from the source documents and to store them in temporary data files, called the source data files. In the second phase the source data are processed to calculate the additional geometric information required for the construction of the geometric model. The third phase comprises the generation of the graphic entities, the development of the topology and the classification of the entities.

Data verification calculations and checks on tolerances are performed throughout the data capture process. The data can then be exported in a format determined by the target GIS software via a data export interface.

4.2.1 The Data Typing Program.

The first phase in the data capture process for the cadastral data is the initial gathering of all the relevant numerical and textual information from the source documents. This is done by using the dedicated data typing program. The typing program has been designed to be used by copy typists and does not require any knowledge of surveying or of cadastral documents. This program is used to enter relevant information gleaned from general plans and diagrams. The data are written to text files or raw data files which serve as the input files to the program modules of the second phase. Functions are provided for the following :

- Coordinates
- Areas
- Lots from General Plans
- Lots from Diagrams
- Connections
- Party Walls.

These functions provide facilities for the automatic incrementation of the lot numbers and the use of constants or default values to limit the required number of keystrokes. A number of independent checks and tests are incorporated in the typing program to eliminate as many typing errors as possible during the typing phase. It is not worthwhile entering the data twice to then compare the two data tables to eliminate typing errors. Typing errors that are not picked up during this phase will become apparent during calculations in the later phases.

The functions used for typing the point coordinates, lot areas, connections and party walls are straight forward. These consist of entering the name or number of the object, such as a point, lot, connection or party wall segment, followed by the numerical values, i.e. coordinates, area or distance and bearing.

The data for each lot are entered by traversing along the lot boundaries. The corner label of the start point, the length and the bearing of the edge or the angle of the corner are entered for each lot boundary. In the case of a diagram which depicts corner angles as opposed to bearings, a bearing must be provided for at least one side of a minimum of one lot in a group of adjacent or interconnected lots. Data is entered in the original units which are used in the source documents. A consistency check is performed during this step to eliminate possible typing errors. A local coordinate system is used for each lot. A closing error of more than 0.02 unit of length is reported and the operator is given the chance of correcting it. If an error cannot be fixed, an option is given to accept the data as a best guess. The consistency check will be executed again during the data import function of the next phase for the attention of a more experienced operator who may have some knowledge of quality control of spatial data acquisition.

Functions are provided throughout to limit the required number of keystrokes and to take advantage of any repetition in the data. Examples of these are the automatic incrementation of lot numbers, the recognition of patterns to anticipate default values for labels and numerical fields for which only the enter key needs to be pressed, and to exploit the repetition of identical lots.

An example raw data file produced by the typing program for cadastral data is given in Appendix E.

4.2.2 The Calculation of the Geometric Information.

The second phase in the data capture process is the calculation of the data required for the geometric model defined in the geometric data base. This phase should preferably be carried out by a person with experience in cadastral calculations. The first step in this phase is to import the source data file produced by the typing program. These data are stored in the workspace file. This data base has a well designed data structure. The consistency checks are repeated for all the lots during the data import step. Any misclosures are reported and the areas are calculated and compared to the values entered from the list of areas. Only after the consistency and area checks have been successfully passed , can the next step be

executed. Thus two quality control checks are incorporated at this stage: the misclosure of the vectors bounding each lot and a comparison of the area calculated from the coordinates or vectors with the area copied from the source document.

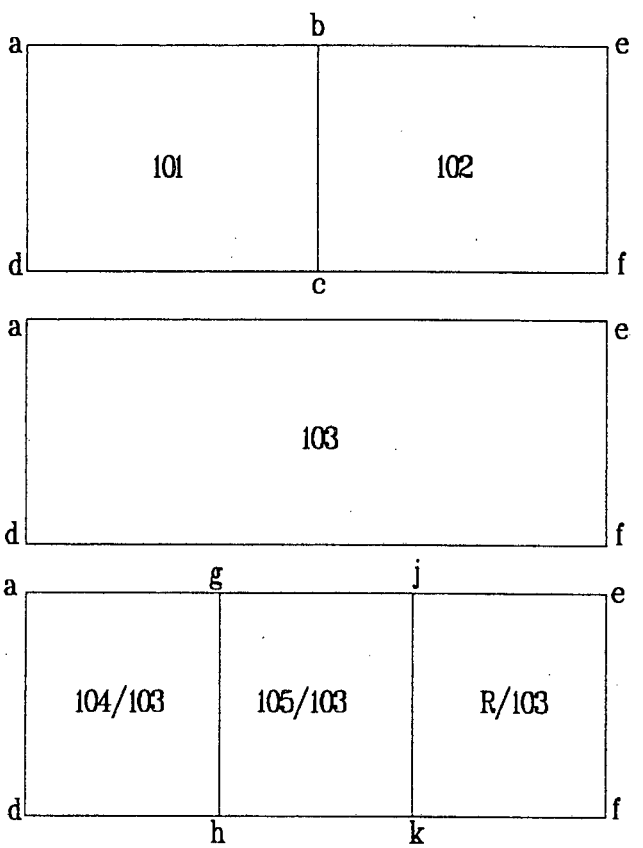
The second step is the calculation of the coordinates for all the lot corners. The original units of measurement are converted to metric units during this step. These calculations are performed during the execution of a number of nested loops. While executing a loop on all the street blocks, or when a specific street block is under consideration, a second loop is executed for all the lots in a block. During execution of this loop on the lots of a block, an inside loop is executed for all the edges of each lot until a lot corner is found for which the coordinates are known in the data base. This can be a fixed point from the source data or the coordinate pair that was calculated earlier during the processing of an adjacent lot. In the case of corner angles being supplied instead of bearings, a check is performed to determine if information is available on the orientation of the lot to calculate the bearings for all the sides using the values of the corner angles. This can be the bearing for an edge as entered or the bearing for an edge as calculated from known endpoints of the edge.

As soon as a starting point with known coordinates is found, the calculations for a traverse along the lot edges are performed and the other lot corners are calculated. A check is performed to determine whether any of these coordinates have been calculated previously. If not, it will be stored in the data base. If the coordinates exist, a test is performed to determine if this new value is within a specified offset or tolerance of the previously calculated value. If so, a weighted average is calculated, otherwise an error is flagged. The loop on all the lots in a block is repeated until all lot corners are calculated. The inside loop on lots is executed in two passes. In the first pass only the original lots are used and in the second pass only the subdivisions. In this way a coordinate of an original lot will not be influenced by a subdivision. On completion of the calculation for a block, a loop on the block corners is executed and the lot corners on the block boundaries are adjusted to fall on line if they are within the specified tolerance, otherwise an error is flagged.

The third step in this phase provides for subdivisions and consolidations. It can become a tedious, time consuming and often complicated task to search for the diagrams reflecting the current status of the land parcels in older parts of towns where many subdivisions and

consolidations have taken place over many years. This step allows for all the diagrams, or the so called full history, to be entered in order that the current status can be generated by the program. Furthermore, diagrams are always framed for subdivisions only and not for the remainders. The data for the remainders are determined by subtracting the subdivisions from the original lots. This procedure is explained with the aid of Figure 4.1.

In Figure 4.1(a) the lots 101 and 102 are represented by the diagrams a.b.c.d and b.e.f.c respectively. These two lots are consolidated into one land parcel, lot 103, represented by the diagram a.e.f.d as shown in figure 4.1(b). This consolidated lot 103 is now subdivided into three portions as shown in figure 4.1(c) where portion 104 of lot 103 is represented by the diagram a.g.h.d and portion 105 of lot 103 is represented by diagram g.j.k.h. The remainder of lot 103 is represented by the diagram j.e.f..k.



(a) Original Land Parcels

(b) Consolidation

(c) Subdivision leaving Remainder 103 and erven 104 and 105 as portions of 103

Figure 4.1 : Consolidation and Subdivision.

4.2.3 Calculations for Graphics and Topology.

At the end of the second phase the data will consists of the coordinates of the corners and the distances and bearings of the edges or boundaries for all current land parcels. These distances and bearings are the dimensions of the lots and can be considered to be the data on the "inside" of the lots. The coordinates and dimensions of the lots are called the diagram data. These data are illustrated for a typical situation in Figure 4.2 (a).

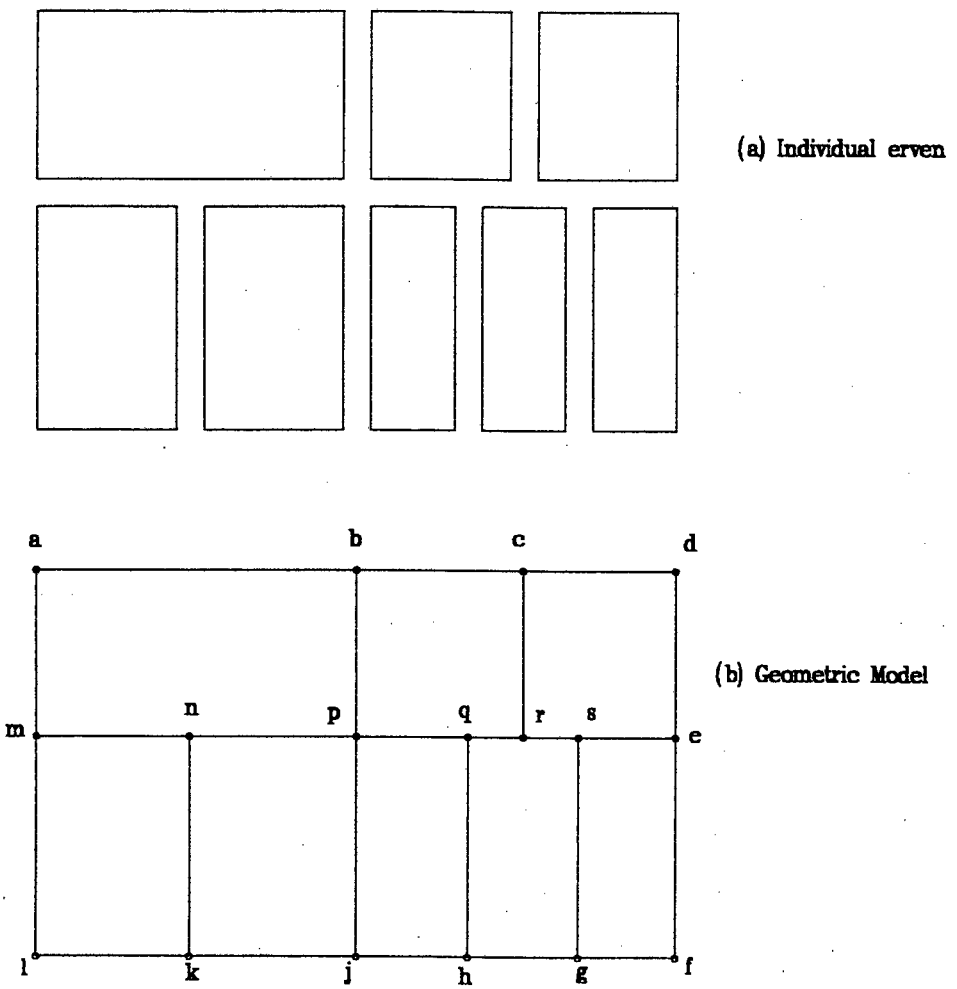


Figure 4.2 : Lot Dimensions and Graphic Lines.

The third phase in the data capture process is the calculation of the required data for the graphic entities that represent the geometric model. The data structure for the geometric model of the cadastral layout makes provision for the development of the topology. This requires the generation of line segments for the lot boundaries so that any segment can only have one land parcel on either side of it. Line segments are generated as illustrated in Figure 4.2 (b). Reference tables are constructed which define the relation between the line segments and the lots or polygons that represent the lots, and as a result the relationship between the lots and the lot corners are defined. The cell reference data structure, as described in the previous chapter, is employed to eliminate the need to continuously search through the complete list of line segments during the construction of the reference tables. For example, the reference table will yield lot 102 as being defined by points b,c,r,q,p and are therefore bounded by line segments b-c, c-r, r-q, q-p, p-b. It will furthermore yield, amongst others, point p as being common to lots 101, 102, 106 and 107. It will also give line segments r-q as being the common boundary between lots 102 and 105 while line segment q-p is the common boundary between lots 102 and 106.

CHAPTER 5.

Data Capture of Engineering Services Data.

5.1 Introduction.

The procedures for capture of the engineering services data follow the same pattern as those of the cadastral data. Procedures are described for the efficient and accurate capture of these data. The preferred sources of data for the engineering services are as-built drawings. However, these are often not available or their validity may be questionable. Field surveys, due to the excessive cost thereof, are often used as a last resort.

5.2 Data Capture from Drawings and Plans.

The most economical sources of data for the engineering services data of a local authority GIS are the as-built drawings. Engineering drawings are not compiled to standardized formats, as are laid down by law for cadastral drawings, but usually the most important information is presented in an easily recognizable form. A methodology has been developed for the capture of data from these documents in an efficient and cost effective way. This is similar to the methods and procedures for the capture of the cadastral data as described in the previous chapter. The procedures are executed in three phases; the Data Typing phase, the Geometry Calculations phase and the Graphics Calculations phase. The computer programs for each phase are designed to be used by an operator with a minimal level of skill. Knowledge or experience in engineering work are only required at specific stages in the process while the rest of the work can be delegated to lower qualified personnel.

In the first phase, a special data typing program is used to gather the data from the source documents and to store them in temporary data files, called the source data files. In the second phase the source data are processed to calculate the additional geometric information required for the construction of the geometric model. The third phase

comprises the generation of the graphic entities, the development of the topology and the classification of the entities.

Data verification calculations and checks on tolerances are performed during the whole data capture process. The data can then be exported in a number of formats as determined by the target GIS.

5.2.1 The Data Typing Program.

The first phase of the capture of the engineering data is the initial gathering of all the relevant information from source documents. As with the cadastral data capture, a dedicated typing program has been designed to be used by copy typists and does not require knowledge of engineering work. It is used to enter all the relevant numerical and textual information that appears on the as-built drawings or design drawings.

Information for the water reticulation networks is usually given on plan layout drawings. The electrical distribution networks are usually also on plan layouts and additional information can often be found on schematic diagrams. Design drawings or as-built drawings for sewer and storm water networks are often given as longitudinal sections for the pipe lines in addition to the plan layouts. However, these designs are sometimes only available on a layout plan with the manhole and pipe information in tabulated form next to the entity positions.

The data is written to text files or raw data files which are the input files to the program modules of the second phase. Functions are provided for the following :

- Sewers
- Water reticulation
- Storm water drainage
- Low tension electric reticulation
- High tension electric reticulation
- Street lights

The use of the functions for the typing of the various services is straight forward. This consists of entering a name or number for a node, i.e. a manhole, a bend, a valve etc., followed by the data items which describe the node. Examples of node data are coordinates, ground level, type, material, diameter, invert level, construction etc. This is followed by the identification and data for the pipe or cable segment which joins the node under consideration to the next one. Examples of this are distance, grade, bearing, material, pipe diameter, cable size etc. These functions provide facilities to limit the required number of keystrokes, e.g. the automatic incrementation of node numbers, the use of constants or default values and exploiting repetition of identical segments. As in the case of the cadastral data, it is not worthwhile entering the data twice and then comparing the two data tables in order to eliminate typing errors. This is also not an effective quality control procedure as the checks are not independent. Any typing errors will become apparent in the calculations which follow in the next phases using the methodology built into the programs.

The co-ordinates for all the nodes of the various engineering services are required by the data capture system. It is required for, amongst other reasons, the calculation of distances and bearings which are used as checks against the values as entered from the source documents. The coordinates are often not given on the engineering design drawings. It is then necessary to calculate the coordinates for the node positions. This is easily done as the positions for the engineering services are always given relative to the cadastral layout of the lots.

The data for the engineering services are entered by travelling along the branches or loops of the network under consideration. The data for a node and the link to the next node are entered for each consecutive branch or loop segment. Consistency checks on depth, grade, bearing, closure etc. are performed during this step to eliminate possible typing errors. Errors greater than pre-defined tolerances are reported and the operator is given the chance of correcting them. If the error cannot be corrected, an option is given to accept the data as the consistency check will again be executed during the data import function of the next phase.

An example raw data file for a sewer drainage network produced by the typing program is given in Appendix F.

5.2.2 The Calculation of the Geometric Information.

The second phase in the data capture process is the calculation of data for the geometric model defined in the geometric data base. This phase should preferably be carried out by someone who has had some experience in the geometry of engineering services. The first step in this phase is to import the source data file or files produced by the typing program. These data are stored in a workspace file. The consistency checks are repeated for all the nodes and segments during this data import step and any misclosures are reported. Only after all the consistency errors have been eliminated, can the next step be executed.

The second step is the calculation of all the node coordinates. These calculations are performed during a loop on all the branches or loops of the network. All the necessary calculations are executed to generate the geometric model. Node coordinates and segment lengths are adjusted to tie in to the cadastral base map as defined by a number of specified connections. Calculated nodes are tested for position on a straight line or join between two specified nodes, and are shifted to fall on the line if they are within specified tolerances. Specified nodes have fixed coordinates which cannot be adjusted.

CHAPTER 6.

Data Capture of Improvements.

6.1 Introduction.

It often is a requirement in a local authority to include information on the extent and position of all the buildings and improvements on the land parcels in the GIS. This can be combined with the valuation records and owners' information to provide a valuation roll on the GIS.

6.2 Required Lot Information.

The assessment office or buildings departments at a local authority may require the building plans to be incorporated into the GIS. One solution to this is to scan all the building plans and to link these scanned images to each lot. This, however, requires a tremendous amount of disk space for the files containing the scanned images.

A more realistic solution is to capture only the most important information from the building plans. This can be the building lines on the lot, the position and outline of the buildings and improvements, the positions of the connection points for the water, electricity and sewer services and the position of the street entrances.

The data for the connection points of the engineering services can usually be obtained from the engineering designs or as-built drawings. This is part of the data capture of the engineering services data as discussed in the previous chapter.

6.3 Building Positions and Outlines.

An efficient and accurate data capture procedure has been developed for the positions and extents of the buildings and improvements. This procedure entails the digitizing of the site plan from the valuation records or building plans using a dedicated digitizing program.

This program employs a number of techniques to reduce the dependency on operator skill and accuracy.

A prerequisite for this procedure is the availability of the cadastral data as described in Chapter 4. The first step in the procedure is to identify the lot by its number and to digitize the lot corners. The ground coordinates of the lot corners will be retrieved and the program will use these values together with the digitized coordinates to calculate the transformation matrix for a Helmert transformation. The Helmert transformation is a least squares transformation. The sum of the squares of the differences is displayed for each transformation matrix, i.e. for each lot. This gives a statistical and rigorous indication of the potential accuracy that can be expected for the digitizing of improvements on each lot.

The second step is to digitize the buildings or improvements. This is done by digitizing the corner points while traversing around the polygon for each individual portion of the building or improvement. A code is also entered to identify the portion, for example dwelling, garage, stoep, store room, car port etc. The Helmert transformation is applied to each digitized point and the ground coordinates are calculated for each point.

The data structure for the storage of the data for the improvements follows a similar hierarchy as for the case of the cadastral data. The data items at the various levels of the data hierarchy are the lots, the polygons for the buildings or improvements, the edges of the improvement polygons and the endpoints of the edges or vertices of the polygons. The relations between the data items at the various levels are maintained in a set of reference tables. Therefore, it is possible to refer to a lot number and retrieve, via the reference table, a list of all the improvement polygons for the lot. A list of all the edges for each improvement polygon can also be obtained via a second reference table. Similarly, a list of all the vertices for each improvement polygon can be obtained via a third reference table.

The data records for the cadastral data of the lots are stored in books 8 and 9 as described in section 3.4 of Chapter 3. The polygons for each portion of the improvements are stored in book 81 and a reference table is defined to relate each lot to the polygons for the improvements. The data records for the edges are stored in book 82 and the data records

for the vertices or corner points are stored in book 83. Reference tables are defined to relate each improvement polygon to its edges and its vertices.

Additional calculations are performed in a third step to further enhance the digitized data. All the digitized points are compared and the points within a certain tolerance are taken as successive digitized values of the same point. Final average values are calculated for these points. All vertex angles within a specified tolerance of ninety degrees are forced to be right angles. All line bearings that are within a given tolerance of each other, including the lot boundaries, are forced to be parallel. The areas of each portion of the buildings and the edge lengths are calculated The calculated areas and edge lengths can be displayed and can be compared to given values from the source documents as a check on the accuracy of the digitizing. The digitizing of a lot can be repeated if the calculated values from the digitized coordinates differ substantially from those given in the source documents.

CHAPTER 7.

Applications of the Data Capture Procedures.

7.1 Introduction.

The methodology for data capture as described in the previous chapters has been developed over a number of years. It has evolved into a sophisticated but practical set of tools which caters for all the aspects of data capture for the cadastral data and the engineering services data known to the author.

7.2 Cadastral Data Capture.

The procedures and programs for the capture of cadastral data have evolved over a period of more than five years and the data for more than 400 000 land parcels have been captured using this system.

The development of the data capture procedures and related computer programs for cadastral data was started after a call for tenders by the municipality of Pretoria for cadastral data capture in the late 1980's. The data for approximately 75 000 lots were captured during which time the system underwent extensive development. Other large data capture projects that were undertaken since were approximately 125 000 lots for the Borough of Durban, 100 000 lots in Soweto and 15 000 lots in Krugersdorp. The programs were furthermore used for cadastral data capture during a number of smaller projects as well as for data verification and consistency checks for land surveyors' general plans on numerous occasions.

The data are captured in the original units and the system will convert these to meters for length or distance and square meters for area. Units of length that are provided for are meters, Cape feet, yards, inches, roods, chains, links and English feet. Units of area that are provided for are square metres, morgen, square roods, square Cape feet, acres, purchases, and square English feet. Provision is made for servitudes and lease areas. A servitude can

be specified as an area or a line with a width. Splays are also catered for. As a rule the sides of the land parcels are straight lines but the system can cater for irregular boundaries and circular arcs. Irregular boundaries are digitised as a large number of short line segments. The data capture is taken directly from the general plans and diagrams. A certain amount of preparatory work must be done before the data can be typed in. This entails the numbering of all the lot corners. This has to be done very carefully as duplicate numbers will cause errors during the processing phase of the procedure. The system will pick up the errors but this can cause unnecessary delays.

The rate at which the data capture can be executed depends on the quality of the source documents. The general plans for new townships are always accurate and legible and data capture can proceed at a fast rate. Older general plans, especially if they were drawn by hand, can be illegible at times. This may then require additional work to perform the necessary calculations in order to reconstruct those parts of the general plans or diagrams. This is time consuming and usually causes considerable delays. The data capture from general plans is also much faster than from diagrams. This is due to a number of factors, of which the most important is that it requires more time to evaluate and plan the task, to number the lot corners and then to type all the alpha numeric data when a large number of documents have to be handled instead of a limited number of drawing sheets showing all the land parcels in relation to each other.

Average rates for data capture of the cadastral data are between 250 and 300 land parcels per man day. These rates are for the typical situations of clear and legible general plans containing at least 85% of the land parcels and data for less than 15% of the lots to be captured from diagrams. Very difficult situations, with the data for a large percentage of the lots to be captured from diagrams with many consolidations and subdivisions, can cause a worst case scenario of less than 50 land parcels per day but these cases are few and far between. In a best case example a rate of more than 500 land parcels per man day was sustained for almost four months. This was achieved during a data capture project of approximately 100000 land parcels in Soweto. The data for all the land parcels were documented on General Plans which were compiled by surveyors during the lease hold surveys of the black townships during the late 1980's. These General Plans were drawn to a

very high standard, either on CAD systems or where all the text was produced by mechanical scribers or stencilled.

Data are transferred to the commercial GIS installations via specified formatted files. A special interface can be developed when a client specifies an unique format. This has been done in the case of the Pretoria and Durban projects referred to above. Interfaces have been developed and used successfully for a number of commercial GIS programs. These are Regis, AllyMap and MapInfo.

Comparative figures for data capture of cadastral data by the "traditional" CAD based methods are, in the author's experience, up to 200 lots per day for a good CAD operator for a normal situation. The same problems concerning older illegible documents and large numbers of diagrams for subdivisions and consolidations apply to CAD drafting methods. However, as pointed out in chapter one, a CAD operator is more expensive, a lot of work remains to be done before a CAD drawing is converted to "intelligent" GIS data and there is not the same control, checks and verification on the consistency and accuracy.

7.3 Engineering Services Data Capture.

The programs and procedures for the capture of engineering services data have been developed over the past three years. To date, the demand for data capture of engineering services has been lower than for cadastral work. Indications are, however, that this will change due to the many GIS installations at local authorities that are maturing beyond the initial start-up phases. The application of the procedures for the capture of engineering services, therefore, has been very limited. The data for the engineering services for approximately 50 000 land parcels have been captured in this way and have proven to be very successful. Further use and the gaining of more experience will no doubt lead to more improvements and refinements to the procedures and programs.

7.4 A Low cost GIS for Local authorities.

Small municipalities often have limited requirements for the storage and easy retrieval of all the data pertaining to a lot or erf. This can be the cadastral information for the lot corners

and lot boundaries, the position and depth of the engineering services, the position, size and orientation of the buildings and improvements, the owner's name and address and the valuation on the land and improvements. The costs involved in establishing and maintaining a commercial GIS may be considered to be prohibitive for a small municipality. A low cost solution can be offered by using one of a number of popular CAD packages which have the ability to couple the graphic entities to the data entries in a set of standard data base files.

The data capture programs described in this thesis also have all the functionality of such a storage and retrieval system. A low cost system, which provides a solution to these minimal requirements, has been installed at a small local authority, namely the municipality of Langebaan. The system is based on a number of the data capture program modules with the addition of a couple of procedures and functions for report generation and some interactive query modules. An important aspect of putting together such a low cost solution is that the data itself are the most valuable asset and care should be taken that a migration path to a more sophisticated GIS is created. It is important to note that once data have been correctly captured, and provided that they are well maintained, they are a valuable asset and care should be taken to ensure that it will be possible to move the data to any other computing platform.

CHAPTER 8.

Closure.

8.1 Conclusions.

The initial data capture for the application of GIS in a local authority is a major project. It is without doubt the single most time consuming and costly aspect of the start-up phase of a GIS at a local authority and can represent 60% or more of the total start-up cost. Special techniques are required if the data capture for a geographic information system is to be done effectively, accurately and efficiently.

There are a number of similarities in the procedures for the data capture of cadastral information and the data capture for the building control and assessment functions and engineering services. The nature of the available source documents for cadastral data, and the checks that can be done to ensure consistency, make it possible to perform this data capture task accurately and effectively. Source documents for engineering services, however, are often not readily available, are often inaccurate or are often non-existent. Costly field surveys may then be the only available alternative. Furthermore, the available documents for engineering services are usually not in the same or a standardised format or do not adhere to the same conventions whereby information is communicated. These inconsistent documents can result in additional paper handling and more time consuming data typing. This can have an influence on the costs of the data capture of the engineering services and/or lead to less control over the accuracy than with the cadastral counterparts. In practice, however, it is acceptable, and the most economical alternative, to start with an accurate cadastral backdrop and provide the information for the engineering services as the best that is available. The accuracy can then be verified and improved during routine inspection and maintenance activities.

The methodology, procedures and techniques for data capture that are described in this thesis have been proved to be effective and efficient on a number of large data capture projects.

8.2 Recommendations.

The methodology, procedures and techniques for data capture that are described in this thesis, are continuously adapted and improved to cater for new requirements and changing circumstances. New techniques are regularly evaluated in order to improve the procedures in terms of accuracy, time and cost.

The data handling scheme, called the record data system or RDS, is central to many of the techniques and algorithms employed in the data capture procedures. Improving the RDS will improve the working of the whole data capture system. The following is a list of possible topics for further research and development work with respect to the RDS :

- Determine the optimum page size and optimum number of pages in the memory map.
- Investigate alternative page-swopping algorithms.
- Provide additional field types for the data records, for example date and memo fields.
- Provide additional storage schemes or data structures, for example queues, trees and graphs.
- Investigate alternative searching and sorting algorithms.

Further development work in these areas will enhance the RDS. This will in turn lead to faster processing and provide more flexible and powerful programming tools.

The data capture procedures can be improved in two ways. Firstly, those activities that currently represent bottle-necks in the data capture process must be improved or eliminated. Examples of two such areas are the manual numbering of the stand corners and the batch mode execution of the programs. The manual numbering of the lot corners causes human errors which, although they are easily identified, leads to time consuming delays in eliminating them. The programs currently executes mainly in batch mode. This is a specific design aspect and will remain as such but the provision of interactive data debugging functions at certain stages of the processing will greatly enhance the system.

Secondly, new techniques or products may replace or enhance existing functions. The following is a list of possible topics for further research and development work with respect to the data capture procedures:

- The elimination of the manual numbering of documents before typing of the data. This will eliminate time consuming human errors.
- The incorporation of improved data structures to improve processing speed..
- Additional and improved error checking schemes for engineering services data.
- The use of scanning technology and vectorization and optical character recognition software.
- The use of pattern recognition techniques.
- Interactive debugging procedures.
- The use of global positioning systems (GPS) as a means of data input during field surveys.

8.3 Summary.

A Geographic Information System in a local authority or municipality is concerned mainly with the information relating to land use and the engineering services infrastructure. The initial data capture, especially for this type of application of GIS, is a major task. It consumes a substantial portion of the budget for the start-up phase in the implementation of a GIS. This is often not appreciated and underestimated in terms of time and costs. There are numerous examples of GIS sites at local authorities which are slow to deliver the expected benefits and results due to the long time taken in becoming operational.

The normal data import and editing functions of most GIS's do not provide the most efficient means for data capture. Special procedures and computer programs have been developed for the data capture of the cadastral layout and municipal services for a GIS. This methodology has proved to be very cost effective and can fulfil the accuracy requirements for a GIS in local authorities.

Data are captured from the best available source documents. Examples of these are surveyors' general plans and diagrams, and engineering as-built drawings or design

drawings. Numerous checks and controls are incorporated into the procedures to ensure that the data will comply with the set accuracy requirements.

The major benefit of this methodology is that highly skilled operators are not required, quality controls are maintained throughout the process and that an accurate topological data base is in place at low cost prior to the purchase of GIS hardware and software. This provides the local authority with the capability to work on the latest GIS technology with a reliable data base once the GIS becomes operational.

REFERENCE APPENDIX.

Anonymous, 1989. ***Summary notes on the conference proceedings.*** News letter of the Hydrological Research Unit, University of Natal, November 1989. (EDIS/SAGIS'89, Earth Data Information Systems / South African Geographic Information Systems Conference, Pieter Maritzburg, July 1989.)

Bettes, A, 1977. ***A data structure for finite elements.*** Int. J. Num. Meth. Engng., Vol.11, 1977.

De Villiers, J F C, 1994. ***Geographical information systems for medium-sized local authorities.*** J. Institution of Municipal Engineers of Southern Africa, Vol.19, No.11, 15-21.

De Wet, F J, 1991. ***The capturing of physical data for GIS at local authorities.*** In EDIS/SAGIS'91, Proceedings of the Earth Data Information Systems / South African Geographic Information Systems Conference, Pretoria, July 1991.

De Wet, F J, 1993. ***Project plan for the implementation of a geographic information system in a local authority.*** Project Report, School of Engineering Management, University of Cape Town, September 1993.

Dickenson, H J and Calkins, H W, 1988. ***The economic evaluation of implimenting a GIS.*** Int. J. Geographic Information Systems, Vol.2, No.4, 307-327.

Foley, J D and Van Dam, A, 1983. ***Fundamentals of interactive computer graphics.*** Addison-Wesley Publishing Company, Inc., 1983.

Lewis, T G and Smith, M Z, 1976. ***Applying data structures.*** Houghton Mifflin Company, 1976.

Labuschagne, F, 1993. ***The effective application of GIS technology in an uncertain environment.*** In SACAD'93, Proceedings of the GIS session at Computer Graphics '93 Conference, Kempton Park, September 1993.

MacDevette, D R, 1991. ***Introducing a GIS: Implications in terms of organisational strategy, structure and culture.*** In EDIS/SAGIS'91, Proceedings of the Earth Data Information Systems / South African Geographic Information Systems Conference, Pretoria, July 1991.

Newman, W M and Sproull, R F, 1979. ***Principles of interactive computer graphics.*** Mcgraw-Hill, Inc., 1979.

Prinsloo, A, 1991. ***The cost of cadastral databases for municipalities.*** Computer Graphics, Vol.2, No.6, 20-21.

Smith, D W and Tomlinson, R F, 1992. ***Assessing costs and benefits of geographical information systems: methodological and implementation issues.*** Int. J. Geographical Information Systems, Vol.6, No.3, 247-256.

Van Rensburg, J, 1990. ***Geographic information systems for urban management versus natural resources management.*** In SACAD'90, Proceedings of the GIS session at Computer Graphics '91 Conference, Kempton Park, September 1990.

Van Rensburg, J and Dickenson, S, 1991. ***Integration of technical information systems within local authorities.*** In EDIS/SAGIS'91, Proceedings of the Earth Data Information Systems / South African Geographic Information Systems Conference, Pretoria, July 1991.

APPENDIX A.

Page Swopping.

A.1 Introduction.

The paging algorithm or page swopping algorithm is the rule that determines which page location in the memory map has to be cleared to receive a page from the random access file during a data record access operation. There are numerous possibilities which vary from simple and fast to sophisticated but computationally more intensive. Various factors are taken into consideration in the different algorithms.

A.2 A Simple Page Swopping Algorithm based on Division.

Probably the most simple algorithm to determine a memory location for a data page is to use a simple division. In this scheme the memory location is given by the remainder when dividing the block number by the number of page locations in the memory map. Mathematically this is given by *pagpos = MOD(blockno,nompag)*, where *pagpos* is the memory location, *blockno* is the block number in the random access file and *nompag* is the number of page locations in the memory map.

For example, if page number 23 of book number 7 resides in block number 314 of the random access file and if the memory map contains copies of 20 pages, then this page will always reside in location 14 of the memory map, i.e. MOD(314,20).

Note that this scheme does not take into consideration factors like frequency of use or time in memory since last access. It furthermore does not distinguish between data pages and pointer pages nor does the book number or page number have any effect. The situation can arise that data records are simultaneously needed from two or more tables which are stored in separate books. It may now happen, with a simple scheme like this one, that the data pages of the books concerned or even the data pages and pointer pages of the same book

will compete for the same memory map location. This will result in unnecessary page swopping in a limited number of memory map locations while a large number of memory map locations remains unused. This is called thrashing.

A number of schemes can now be used in an effort to eliminate potential thrashing. One such scheme, which uses the calculation of different offsets for the different book numbers as well as an offset for the pointer pages, were used in an earlier version of the RDS system. This was later abandoned in favour of a more sophisticated paging scheme.

A.3 A Sophisticated Page Swopping Scheme.

A page swopping scheme is needed that will take into account factors like frequency of use and time in memory. It is easy to realise that a page which is accessed often should remain in memory and that a page that is used less frequently would be a more suitable candidate for page swopping. However, monitoring frequency of use alone will discriminate against pages that have only recently been brought into memory and which might be needed often hence forth. A scheme has been devised whereby time in memory as well as frequency of use is monitored.

Two parameters are used in this paging algorithm. These parameters, called *tim* and *lru,* are maintained for each memory map location and are kept in the memory map index, The parameter *tim* is used to reflects 'time in memory', and is a simple counter. The parameter *lru* reflects the 'least recently used' page, and is a variable with a large initial value which dies away with time in memory. Every time a data record is accessed from a page in the memory map, the parameter *lru* for that page is incremented by a large number while the parameter *tim* for every other page in the memory map is incremented by one. The value of the parameter *lru* for all page locations are devided by two after a fixed number of accesses to any page. This will let the value of *lru* die away with time in memory when a page is not accessed and prevents a numeric overflow when a page is constantly accessed for a large number of times. When a data page is required of which a copy is not in the memory map, that page location with the smallest value for *lru* and the largest value for *tim* is the most suitable candidate for a page swop.

This algorithm gives excellent results and does not require excessive calculations. It is independent of the book number and does not distinguish between data pages or pointer pages. It eliminates thrashing and minimises disk input/output operations because the only consideration is actual frequency of use and time in memory for any data block.

APPENDIX B.

Hashing.

B.1 Common Techniques.

The Scheme of key-to-address transformation is commonly called hashing. The more common transformations are the following:

1. Division: The key is divided by an integer number (often a prime number) slightly smaller or equal to the number of distinct addresses available. The remainder of the division is used as the address.

2. Midsquare: The key is squared and the digits in the middle are retained for the address.

3. Folding: The key is divided into several parts, each of which has no more digits than the desired address. Then a mathematical operation is performed on the parts. Folding offers a variety of randomising techniques, depending upon the operation chosen. For example, the parts could be added together.

4. Random: The key is used as input to a random number generator. The output of the generator will be the address.

Ideally the computation performed on a key would yield unique addresses for different keys and scatter the records uniformly throughout the address space. Because of this, the term *scatter storage* is applied to the technique, as well as randomising, hashing, or key-to-address transformation.

B.2 Hashing in the RDS.

The RDS system allows alpha-numeric labels for data records. The key-to-address transformation is done by using the ASCII value for each character in the key or label field. A very simple algorithm was used initially which calculated the sum of the ASCII values and then applying the division technique. This is given as follows:

```
sum = 0
for i = 1 to length
        val = ASC(label(i))
        sum = sum + val
end
hash = MOD(sum,space)
```

where *length* is the number of characters in the label or key field, *label(i)* is the i-th character in the label or key field and *space* is the allowable address space.

This algorithm often gave poor results for scatter. The maximum value for the sum of the ASCII values is less than 127 times the number of characters in the key field. This is less than a thousand for eight character labels and even less for shorter labels.

A modified algorithm, using a combined folding and division technique is currently in use and gives excellent results. The algorithm is as follow:

```
for i = 1 to length
        val(i) = ASC(label(i))
end
sum = 0
mult = 1
i = 1
j = length
```

```
do while i < j
        sum = sum + (label(i) + label(j)) * mult
        mult = mult * 10
        i =i + 1
        j = j - 1
enddo
if i = j then
        sum = sum + label(i)
endif
hash = MOD(sum,space)
```

The folding technique using the first plus last, the second plus second last, and so on, of the characters in the labels eliminates the sensitivity to the length of the label. The multiplication of the pairs with increasing factors of 10 generates a large pseudorandom number which, together with the division, results in a well scattered address transformation.

APPENDIX C.

Line - Cell Relations.

C.1 Geometric representation of Lines.

The geometric representations of straight-line segments or line objects are described by the co-ordinate pairs of the two end points. The raster or pixel representation of a line reduces to the computation of the co-ordinates of the pixels which lie near the line on a two-dimensional raster grid.

C.2 Bresenham's Line Algorithm.

An interesting line-drawing algorithm has been developed by Bresenham. (Newman and Sproull, 1979, and Foley and Van Dam, 1983). It is designed so that each iteration changes one of the co-ordinate values by plus or minus one. The other co-ordinate may or may not change, depending on the value of an error term maintained by the algorithm. This error term records the distance, measured perpendicular to the axis of greatest movement, between the exact path of the line and the actual pixel generated. This is illustrated in figure C.1, where the x-axis is the axis of greatest movement and the error term e is shown measured parallel to the y-axis. The following description of the algorithm assumes this particular orientation of the line.

At each iteration of the algorithm the slope of the line, dy/dx, is added to the error term e. Before this is done, the sign of e is used to determine weather to increment the y co-ordinate of the current point. A positive e value indicates that the exact path of the line lies above the current point: therefore the y co-ordinate is incremented, and 1 is subtracted from e. If e is negative, the y co-ordinate value is left unchanged. The basic algorithm is expressed in its simple form and a modified form, as follows:

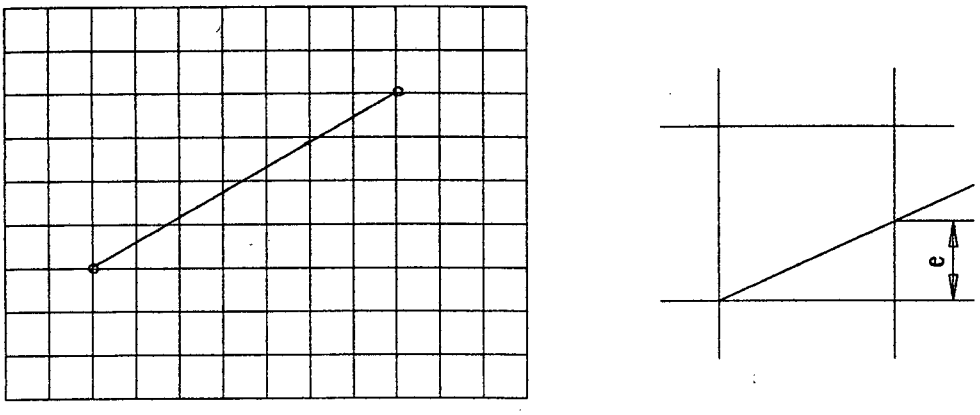


Figure C.1 : Bresenham's Pixel representation of a Line.

```
e = dy/dx - 0.5
for i = 1 to dx
        Plot(x,y)
        If e  >  0 then
                y = y+1
                e = e + dy/dx - 1
        else
                e =  e + dy/dx
        endif
        x = x + 1
end
```

```
e = 2dy - dx
for i = 1 to dx
        Plot(x,y)
        if e > 0 then
                y = y + 1
                e = e + 2dy - 2dx
        else
                e = e + 2dy
        endif
                x = x + 1
end
```

The function Plot(x,y) sets the pixel at co-ordinate position (x,y).

In the modified form, the error term was multiplied by 2dx. This eliminated the division and reduces the calculations to simple integer additions and subtractions. The error test is still valid because the test relies only on the sign of the error term.

C.3 The Reference Cell Algorithm.

The Bresenham algorithm can be utilised to determine the raster cells that are traversed by a straight line. The increments in the x and y co-ordinates indicate whether the next cell is the row direction or the column direction. The only modification to the algorithm is in the initial value for the error term e, as is explained by the diagram of figure C.2.

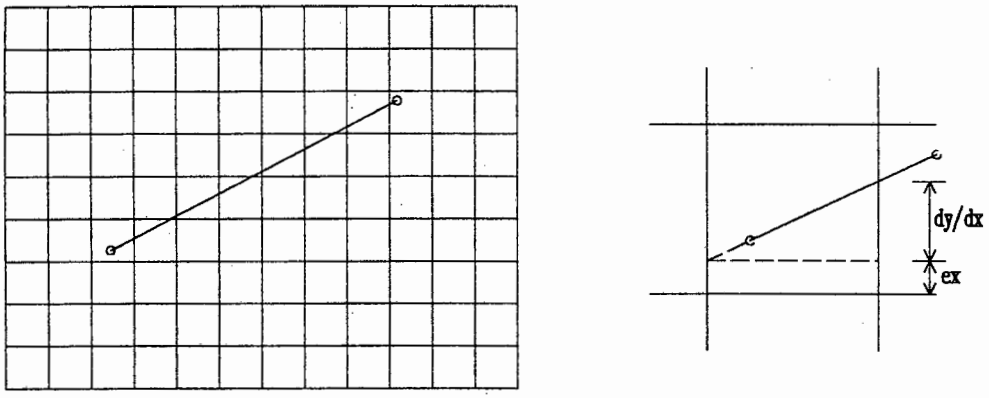


Figure C.2 : Raster Cells traversed by a Line.

Using the modified algorithm, the cell numbers can be calculated as follows:

```
e = ex + 2dy - dx
j = j1
for i = i1 to i2
        Cell (i,j)
        if e > 0 then
                j = j + 1
                e = e + 2dy - 2dx
        else
                e = e + 2dy
        endif
```

$i = i + 1$

end

The function Cell(i,j) returns the cell number at the intersection of row i and column j.

This algorithm generates the sequence of cell numbers of the raster grid cells that are traversed by the straight line. A full implementation of the algorithm involves allowing for other cases besides $0 \leq dy \leq dx$ as outlined above.

In practice, three special cases are tested for before applying the algorithm. These are the cases where the whole line falls in a single cell, a single row of cells or a single column of cells.

APPENDIX D.

Polygon - Cell Relations.

D.1 Geometric representation of Areas.

The geometric representations of area objects are described by the boundaries or outlines. For a polygon, the outline can be represented by an ordered list of vertices. Although it does not matter which vertex is listed first, the vertices are ordered so that adjacent vertices in the list represent edges of the polygon. The last vertex is, by convention, adjacent to the first vertex. The ordered list of vertices implies an ordered list of edges of the polygon.

D.2 Scan Converting Polygons.

The scan conversion of polygons for a raster display device involves systematically scanning all the pixels to find those pixels that lie inside the polygon boundaries. A display of the polygon is generated by appropriately setting the inside pixels as opposed to the outside pixels. From a description of the outline of a polygon it is possible to decide weather a given point lies inside or outside the polygon by counting intersections of the boundary with an imaginary line extending from the point to some other point far outside the polygon. If an odd number of intersections is encountered, the point lies inside the boundary; otherwise it lies outside.

These observations suggest a simple scan-conversion algorithm to find the raster grid cells that are overlaid or partially overlaid by a polygon. A cell is considered related if any one of the four corner points lie within the polygon. A function tests to see if a cell corner point (x,y) lies within the polygon by counting intersections of the line from (x,y) to (∞,y) with each edge of the polygon and returning *true* if the count is odd or *false* if it is even. This simple algorithm is potentially unacceptably slow. Testing each cell requires intersecting four test lines with each edge of the polygon. A large polygon or one with many edges will be processed very slowly.

D.3 Coherence.

The performance of the scan-conversion algorithm can be improved substantially by taking advantage of the coherence property. If a given pixel is inside the polygon, immediately adjacent pixels are likely to be inside as well. A similar coherence holds for pixels outside the boundary. The coherence property suggests that a number of adjacent pixels should be tested together. A convenient group to test is an entire scan line. This will yield two or a larger even number of intersections of each scan line with the polygon edges. Each pair of intersection points represents a region of the scan line that lie inside the polygon.

This discussion leads to an algorithm substantially better in performance than the first one. For each scan line:

- Find the intersection of the scan line with all edges of the polygon
- Sort the intersections by increasing x-co-ordinate, and
- Fill in all pixels between pairs of intersections.

A similar Scan-Line algorithm can be utilised to determine the raster cells that are overlaid or partially overlaid by a polygon. The identical algorithm is used with the introduction of one new observation, namely that the intersections can be computed first, and sorted out later:

- For each edge of the polygon, compute all intersections with the scan lines and build a list of intersections
- Sort the list so that intersections for each scan line are grouped together and x values within a scan line increase
- Fill all grid cells between pairs from the list, since each pair represents a region of a scan line inside the polygon.

The application of this algorithm is illustrated with the aid of figure D.1. The first step of calculating all intersections for each edge of the polygon is effectively achieved by determining the grid cell representation of each edge by the modified Breshenham algorithm, as described in Appendix C. This yields a list of cell numbers which are grouped for each row and sorted within each row in the second step. Step three requires the

removal of elements from the list in pairs where each pair represents a region of a cell row inside the polygon.

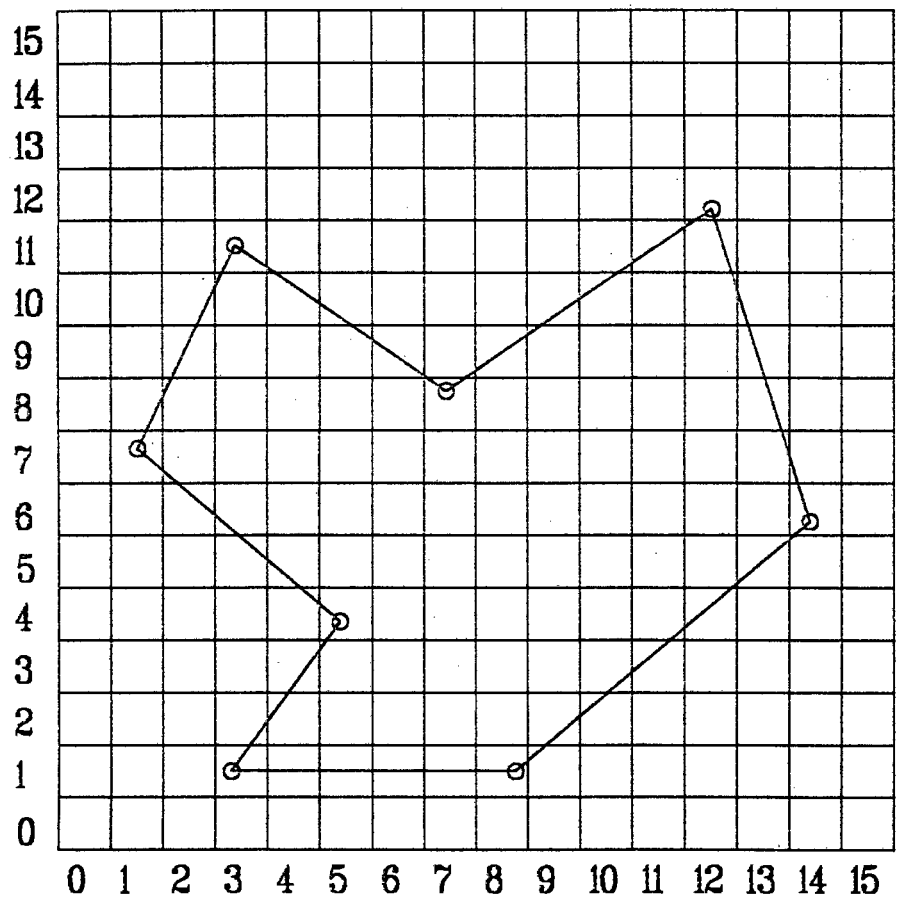


Figure D.1 : Raster Cells overlayed by a Polygon.

Special consideration is required to account for adjacent cells. Cells (1,8) and (2,8) are not a pair, due to the fact that they are overlaid by the same edge, whereas cells (11,11) and (12,11) are a pair, due to the fact that they represent different edges. Special consideration is also required at local minima and local maxima. Cell (3,11) is considered a pair, as is cell (7,8), due to the fact that they represent a local maxima and a local minima respectively. Cell (1,7) and cell (14,6) are not a local minima or a local maxima and therefore does not require special consideration.

Four special cases are also considered. These are when the whole polygon falls in one cell only, in a row of cells, in a column of cells or in a square of four cells.

APPENDIX E.

Example of Cadastral Data File.

The followinf is an example of cadastral data as produced by the cadastral data typing program as described in Section 4.2.1.

```
.COOR     A   23654.670 10818531.040  10 E
.COOR     B   22835.360 10818480.040  10 E
.COOR     C   22813.910 10818812.410  10 E
.COOR     D   22949.660 10818821.210  10 E
.COOR     E   23594.860 10819048.240  10 E
.COOR     F   23708.010 10818915.400  10 E
.COOR    S17    7151.850  3297602.360   8 M
.COOR    S18    7154.690  3297603.360   8 M
.COOR  BA119    7147.880  3297600.970   8 M
.COOR  BA120   23389.670 10818585.150   8 E
.COOR  BA126   23664.560 10818602.290   8 E
.COOR     S1    7155.240  3297455.310   8 M
.COOR     S2    7152.230  3297455.120   8 M
.COOR   BAA1    7157.710  3297455.460   8 M
.COOR   BAA2    7160.850  3297477.170   8 M
.COOR     S3    7156.250  3297476.890   8 M
.COOR     S4    7153.240  3297476.700   8 M
.COOR  BA121    7131.830  3297493.920   8 M
.COOR     S5    7157.120  3297495.490   8 M
.COOR     S6    7154.110  3297495.300   8 M
.COOR     S8    7155.260  3297519.870   8 M
.COOR     S7    7158.270  3297520.060   8 M
.COOR  BA122    7135.400  3297518.630   8 M
.COOR    S10    7135.840  3297521.660   8 M
.COOR     S9    7155.630  3297522.900   8 M
.COOR      5    7139.890  3297549.980   8 M
.COOR   BAA5    7171.590  3297551.940   8 M
.COOR    S12    7162.550  3297551.380   8 M
.COOR    S11    7159.500  3297551.190   8 M
.COOR    S20    7159.840  3297553.730   8 M
.COOR    S14    7160.200  3297568.440   8 M
.COOR    S16    7159.830  3297569.960   8 M
.COOR    S15    7162.850  3297570.230   8 M
.COOR    S13    7163.210  3297568.770   8 M
.COOR    S19    7162.840  3297553.490   8 M
.COOR  BA123   23698.610 10818847.550   8 E
.COOR  BA124   23684.340 10818744.780   8 E
.COOR  BA125   23673.080 10818663.640   8 E
.COOR  BA200   23412.730 10818889.250   8 E
.COOR      1   23312.370 10818509.660   8 E
.COOR      2   23358.130 10818828.380   8 E
.COOR      3   23331.790 10818955.680   8 E
.COOR      4   23394.620 10818977.780   8 E
.COOR      6    7126.010  3297453.480   8 M
```

```
.COOR      7   23334.110 10818661.110   8 E
.COOR      8   23317.700 10818675.480   8 E
.COOR      9   23248.390 10818655.980   8 E
.COOR     10   23059.980 10818644.280   8 E
.COOR     11   23000.410 10818607.520   8 E
.COOR     12   22979.760 10818629.280   8 E
.COOR     13   22976.670 10818679.190   8 E
.COOR     14   22994.470 10818703.340   8 E
.COOR     15   23058.120 10818674.220   8 E
.COOR     16   23243.340 10818685.720   8 E
.COOR     17   23327.170 10818709.320   8 E
.COOR     18   23343.750 10818728.220   8 E
.COOR    SS1    7126.450  3297456.563   7 M
.COOR    SS2    7204.095  3297461.374   7 M
.COOR    SS3    7203.667  3297458.290   7 M
.COOR    SS4    7220.280  3297578.001   7 M
.COOR    SS5    7225.577  3297571.783   7 M
.COOR    SS6    7190.685  3297612.746   7 M
.COOR    SS7    7148.907  3297598.045   7 M
.COOR    SS8    7083.823  3297450.812   7 M
.COOR    SS9    7088.702  3297454.190   7 M
.COOR   SS10    7014.984  3297554.154   7 M
.COOR   SS11    7112.090  3297585.089   7 M
.COOR   SS12    7070.738  3297570.539   7 M
.COOR   SS13    7087.445  3297506.650   7 M
.COOR   SS14    7067.851  3297569.523   7 M
.COOR   SS15    6995.610  3297544.103   7 M
.COOR   SS16    6956.857  3297541.592   7 M
.COOR   SS17    6963.003  3297446.365   7 M
.OPPV   2799      .0000     .0000     .0000  2508.0000 E
.OPPV   2801      .0000     .0000     .0000 11199.0000 E
.OPPV   2802      .0000     .0000     .0000 12957.0000 E
.OPPV   2803      .0000     .0000     .0000 15358.0000 E
.OPPV   2804      .0000     .0000     .0000 12963.0000 E
.OPPV   2805      .0000     .0000     .0000 16953.0000 E
.OPPV   2806      .0000     .0000     .0000 13457.0000 E
.OPPV   2807      .0000     .0000     .0000 15874.0000 E
.OPPV   2808      .0000     .0000     .0000 14845.0000 E
.OPPV   2809      .0000     .0000     .0000 19132.0000 E
.OPPV   2810      .0000     .0000     .0000 14613.0000 E
.OPPV   2811      .0000     .0000     .0000 11262.0000 E
.OPPV   2812      .0000     .0000     .0000 10247.0000 E
.OPPV   2813      .0000     .0000     .0000 10386.0000 E
.OPPV   2800     3.0367     .0000     .0000     .0000 A
.BLOK   1
.KAART 2800          S02392/1958 E    .00    .00    .00     .00
.DATA      6   319.8500    8.1010    8.1010
.DATA      5   110.1000  348.1840  348.1840
.DATA   BA117    65.4100   70.3650   70.3650
.DATA   BA118    40.0000  340.3650  340.3650
.DATA   BA119   152.2800   70.3650   70.3650
.DATA      E   174.6800  139.3200  139.3200
.DATA      F   388.1100  187.5410  187.5410
.DATA      A   275.7600  266.2650  266.2650
```

```
.KAART 1/2800        S03395/1965 E      .0000 .0000  .0000 19428.
.KAART          L S00553/1965 M      .0000
.DATA      6   72.0000    8.1010    8.1010
.DATA   BA120  275.4200   86.2550   86.2550
.DATA   BA126   71.9300  187.5410  187.5410
.DATA      A  275.7600  266.2550  266.2550
.KAART 2/2800        S03396/1965 E      .0000 .0000  .0000 16710.
.KAART          L S00553/1965 M      .0000
.DATA   BA120   62.0000    8.1010    8.1010
.DATA   BA121  275.1300   86.2550   86.2550
.DATA   BA125   61.9400  187.5410  187.5410
.DATA   BA126  275.4200  266.2550  266.2550
.KAART 3/2800        S03397/1965 E      .0000 .0000  .0000 22073.
.KAART          L S00553/1965 M      .0000
.DATA   BA121   82.0000    8.1010    8.1010
.DATA   BA122  274.7400   86.2550   86.2550
.DATA   BA124   81.9200  187.5410  187.5410
.DATA   BA125  275.1300  266.2550  266.2550
.KAART 4/2800        S03398/1965 E      .0000 .0000  .0000 27912.
.KAART          L S00553/1965 M      .0000
.DATA   BA122  103.8500    8.1010    8.1010
.DATA      5  274.2500   86.2550   86.2550
.DATA   BA123  103.7500  187.5410  187.5410
.DATA   BA124  274.7400  266.2550  266.2550
.KAART 5/2800        S03399/1965 E      .0000 .0000  .0000 18040.
.KAART          L S00553/1965 M      .0000
.DATA      5   60.0000  348.1840    8.1010
.DATA   BA200  296.4400   84.5620   86.2550
.DATA      F   68.5100  187.5410  187.5410
.DATA   BA123  274.2500  266.2550  266.2550
.KAART 2799          S02392/1958 E    .00   .00    .00     .00
.DATA      4   60.0000   70.3650   70.3650
.DATA   BA119   40.0000  160.3650  160.3650
.DATA   BA118   65.4100  250.3650  250.3650
.DATA   BA117   40.3600  348.1840  348.1840
.BLOK   1
.KAART 2801          S02392/1958 E    .00   .00    .00     .00
.DATA      1   54.7400  266.2650  266.2650
.DATA   BB101  150.0000  356.2550  356.2550
.DATA      9   72.0000   74.1700   74.1700
.DATA      8   21.8200  131.1330  131.1330
.DATA      7  153.0000  188.1010  188.1010
.KAART 2802          S02392/1958 E    .00   .00    .00     .00
.DATA   BB101   50.0000  266.2650  266.2650
.DATA   BB128  166.7400  330.3310  330.3310
.DATA   BB127  122.7700   86.2650   86.2650
.DATA      9  150.0000  176.2550  176.2550
.KAART 2803          S02392/1958 E    .00   .00    .00     .00
.DATA   BB128  138.7700  266.2650  266.2650
.DATA   BB102  150.0000  356.2540  356.2540
.DATA     10   66.0000   86.2650   86.2650
.DATA   BB127  166.7400  150.3310  150.3310
.KAART 2804          S02392/1958 E    .00   .00    .00     .00
.DATA   BB102  142.4200  266.2650  266.2650
```

```
.DATA   BB103   142.1000    31.0110    31.0110
.DATA      11    70.0000    58.1910    58.1910
.DATA      10   150.0000   176.2540   176.2540
.KAART 2805          S02392/1958 E   .00   .00   .00   .00
.DATA   BB103    92.0000   266.2650   266.2650
.DATA       B   110.0600   356.1830   356.1830
.DATA   BB104   156.5300    75.2500    75.2500
.DATA      12    30.0000   136.3040   136.3040
.DATA      11   142.1000   211.0110   211.0110
.KAART 2806          S02392/1958 E   .00   .00   .00   .00
.DATA   BB104   125.0000   356.1830   356.1830
.DATA   BB105   160.4100   102.4530   102.4530
.DATA      13    50.0000   176.2650   176.2650
.DATA      12   156.5300   255.2500   255.2500
.KAART 2807          S02392/1958 E   .00   .00   .00   .00
.DATA   BB105    98.0000   356.1830   356.1830
.DATA       C    86.0300    86.1730    86.1730
.DATA   BB106   148.7000   140.2620   140.2620
.DATA      14    30.0000   216.2350   216.2350
.DATA      13   160.4100   282.4530   282.4530
.KAART 2808          S02392/1958 E   .00   .00   .00   .00
.DATA   BB106    50.0000    86.1730    86.1730
.DATA       D   100.0000    70.3650    70.3650
.DATA   BB119   180.7300   175.3050   175.3050
.DATA      15    70.0000   294.3430   294.3430
.DATA      14   148.7000   320.2620   320.2620
.KAART 2809          S02392/1958 E   .00   .00   .00   .00
.DATA   BB108   208.1100   174.3550   174.3550
.DATA   BB118   100.0000   266.2650   266.2650
.DATA      15   180.7300   355.3050   355.3050
.DATA   BB119   100.0000    70.3650    70.3650
.KAART 2810          S02392/1958 E   .00   .00   .00   .00
.DATA   BB108    50.7100    70.3650    70.3650
.DATA   BB109   226.0600   165.2050   165.2050
.DATA      16    85.5800   266.2650   266.2650
.DATA   BB118   208.1100   354.3550   354.3550
.KAART 2811          S02392/1958 E   .00   .00   .00   .00
.DATA      16   122.0600   345.2050   345.2050
.DATA   BB113   140.7000    97.2510    97.2510
.DATA   BB116    58.0000   188.1010   188.1010
.DATA      18    25.1400   221.1330   221.1330
.DATA      17    87.0900   254.1700   254.1700
.KAART 2812          S02392/1958 E   .00   .00   .00   .00
.DATA   BB116   140.7000   277.2510   277.2510
.DATA   BB113    42.0000   345.2050   345.2050
.DATA   BB112   149.9800    75.2420    75.2420
.DATA   BB115    55.0000   168.1840   168.1840
.DATA       2    43.1800   188.1010   188.1010
.KAART 2813          S02392/1958 E   .00   .00   .00   .00
.DATA   BB115   149.9800   255.2420   255.2420
.DATA   BB112    62.0000   345.2050   345.2050
.DATA   BB109   154.3900    70.3650    70.3650
.DATA       3    75.0000   168.1840   168.1840
.BLOK   1
```

```
.KAART VD101/2800      S03937/1984 M    101.0000  3.00
.KAART           L S01446/1984 M       .0000
.DATA     S15
.DATA     S16
.DATA     S17
.DATA     S18
.KAART VD6/1/2800      S03938/1984 M     65.0000  3.00
.KAART           L S01446/1984 M       .0000
.DATA     S1
.DATA     S2
.DATA     S4
.DATA     S3
.KAART VD7/2/2800      S03939/1984 M     56.0000  3.00
.KAART           L S01446/1984 M       .0000
.DATA     S3
.DATA     S4
.DATA     S6
.DATA     S5
.KAART VD8/3/2800      S03940/1984 M     74.0000  3.00
.KAART           L S01446/1984 M       .0000
.DATA     S5
.DATA     S6
.DATA     S8
.DATA     S7
.KAART VD9/4/2800      S03941/1984 M    154.0000  3.00
.KAART           L S01446/1984 M       .0000
.DATA     S7
.DATA    BA122
.DATA     S10
.DATA     S9
.DATA     S11
.DATA     S12
.KAART VD10/5/2800     S03942/1984 M     57.0000  3.00
.KAART           L S01446/1984 M       .0000
.DATA     S12
.DATA     S11
.DATA     S20
.DATA     S14
.DATA     S16
.DATA     S15
.DATA     S13
.DATA     S19
.BLOK   52
.KAART VD101/2800      S02392/1958 M    237.0000  3.05
.DATA     SS2
.DATA     SS3
.DATA      6
.DATA     SS1
.KAART VD102/2800      S02392/1958 M    717.0000  6.10
.DATA     SS3
.DATA     SS4
.DATA     SS5
.DATA      A
.KAART VD103/2800      S02392/1958 M    303.0000  3.05
```

```
.DATA     SS5
.DATA     SS6
.DATA     SS7
.DATA    BA119
.DATA      E
.DATA      F
BLOK  2
.KAART VD101/2802      S02392/1958 M   1391.0000  3.05
.DATA     SS11
.DATA     SS12
.DATA     SS13
.DATA      16
.DATA     SS14
.DATA     SS15
.DATA     SS16
.DATA     SS17
.DATA      SS9
.DATA    BB101
.DATA      B
.DATA      C
.DATA      D
.DATA      3
.KAART VX102/2802      S02392/1958 M   1104.0000  9.14
.DATA     SS8
.DATA    BB128
.DATA     SS10
.DATA    BB119
.END
```

APPENDIX F.

Example of Services Data File.

The followinf is an example of a sewer network data as produced by the services data typing program as described in Section 5.2.1.

```
.MANH  UIT2   7.506   0.800  1.050  0.000
.CHAIN VP1    13.660  0.800  1.050  0.000
.CHAIN VP2    12.543  0.800  1.050  0.000  300.0  80.000  0.0
.CHAIN SW1    12.381  0.800  1.050  0.000  300.0  11.531  0.0
.CHAIN VP3    11.564  0.800  1.050  0.000  300.0  55.021  0.0
.CHAIN VP4    10.457  0.800  1.050  0.000  300.0  80.000  0.0
.CHAIN VP5    9.824   0.800  1.050  0.000  300.0  69.002  0.0
.CHAIN SW2    8.937   0.800  1.050  0.000  375.0  68.448  0.0
.CHAIN VP6    9.115   0.800  1.050  0.000  375.0  22.162  0.0
.CHAIN SW3    9.057   0.800  1.050  0.000  375.0  10.909  0.0
.CHAIN SW4    8.679   0.800  1.050  0.000  525.0  42.100  0.0
.CHAIN VP7    8.626   0.800  1.050  0.000  525.0  42.348  0.0
.CHAIN SW5    8.680   0.800  1.050  0.000  525.0  11.050  0.0
.CHAIN SW6    8.517   0.800  1.050  0.000  525.0  29.557  0.0
.CHAIN UIT1   10.365  0.800  1.050  0.000  525.0  13.792  0.0
.CHAIN UIT2   7.506   0.800  1.050  0.000  525.0  69.371  0.0
.CHAIN VP13   14.305  0.800  1.050  0.000  300.0
.CHAIN SW7    14.305  0.800  1.050  0.000  300.0   1.949  0.0
.CHAIN SW8    14.237  0.800  1.050  0.000  300.0  80.000  0.0
.CHAIN SW9    13.245  0.800  1.050  0.000  300.0  60.064  0.0
.CHAIN SW10   13.425  0.800  1.050  0.000  300.0  11.200  0.0
.CHAIN SW11   13.512  0.800  1.050  0.000  300.0  83.002  0.0
.CHAIN SW12   12.563  0.800  1.050  0.000  300.0  80.000  0.0
.CHAIN SW13   10.972  0.800  1.050  0.000  300.0  43.982  0.0
.CHAIN SW14   9.988   0.800  1.050  0.000  300.0  46.992  0.0
.CHAIN SW15   9.004   0.800  1.050  0.000  300.0  16.208  0.0
.CHAIN VP8    8.733   0.800  1.050  0.000  300.0  11.376  0.0
.CHAIN SW6    8.517   0.800  1.050  0.000  300.0  33.896  0.0
.CHAIN VP23   15.317  0.800  1.050  0.000
.CHAIN SW16   15.265  0.800  1.050  0.000  300.0   2.125  0.0
.CHAIN SW17   14.503  0.800  1.050  0.000  300.0  80.002  0.0
.CHAIN SW18   14.671  0.800  1.050  0.000  300.0  18.907  0.0
.CHAIN SW19   13.796  0.800  1.050  0.000  300.0  44.041  0.0
.CHAIN SW20   13.327  0.800  1.050  0.000  300.0  44.954  0.0
.CHAIN SW21   12.527  0.800  1.050  0.000  300.0  80.000  0.0
.CHAIN SW22   11.874  0.800  1.050  0.000  300.0  62.716  0.0
.CHAIN SW23   11.588  0.800  1.050  0.000  300.0  19.563  0.0
.CHAIN SW24   11.218  0.800  1.050  0.000  375.0  26.844  0.0
.CHAIN SW25   10.450  0.800  1.050  0.000  375.0  31.933  0.0
.CHAIN SW26   9.213   0.800  1.050  0.000  375.0  72.753  0.0
.CHAIN SW3    9.057   0.800  1.050  0.000  375.0  15.992  0.0
.CHAIN VP32   14.125  0.800  1.050  0.000  300.0
.CHAIN SW30   14.070  0.800  1.050  0.000  300.0   2.125  0.0
.CHAIN SW31   12.718  0.800  1.050  0.000  300.0  80.012  0.0
.CHAIN SW1    12.381  0.800  1.050  0.000  300.0  16.210  0.0
```

```
.CHAIN VP21    10.974  0.800  1.050  0.000  300.0
.CHAIN SW27    10.815  0.800  1.050  0.000  300.0   1.875  0.0
.CHAIN SW28    10.756  0.800  1.050  0.000  300.0   9.522  0.0
.CHAIN SW29    9.081   0.800  1.050  0.000  300.0  76.906  0.0
.CHAIN SW5     8.680   0.800  1.050  0.000  300.0  16.208  0.0
.CHAIN VP14    13.323  0.800  1.050  0.000  300.0
.CHAIN SW8     13.245  0.800  1.050  0.000  300.0   1.949  0.0
.CHAIN VP15    13.522  0.800  1.050  0.000  300.0
.CHAIN SW9     13.425  0.800  1.050  0.000  300.0   1.950  0.0
.CHAIN VP16    13.655  0.800  1.050  0.000  300.0
.CHAIN SW35    13.594  0.800  1.050  0.000  300.0   1.950  0.0
.CHAIN SW10    13.512  0.800  1.050  0.000  300.0  18.451  0.0
.CHAIN VP17    12.641  0.800  1.050  0.000  300.0
.CHAIN SW11    12.563  0.800  1.050  0.000  300.0   1.950  0.0
.CHAIN VP18    11.025  0.800  1.050  0.000  300.0
.CHAIN SW12    10.972  0.800  1.050  0.000  300.0   1.950  0.0
.CHAIN VP19    10.198  0.800  1.050  0.000  300.0
.CHAIN SW13    9.988   0.800  1.050  0.000  300.0  10.929  0.0
.CHAIN VP20    9.066   0.800  1.050  0.000  300.0
.CHAIN SW14    9.004   0.800  1.050  0.000  300.0   2.125  0.0
.CHAIN VP22    9.148   0.800  1.050  0.000  300.0
.CHAIN SW29    9.081   0.800  1.050  0.000  300.0   1.800  0.0
.CHAIN VP25    14.686  0.800  1.050  0.000  300.0
.CHAIN SW36    14.622  0.800  1.050  0.000  300.0   1.875  0.0
.CHAIN SW18    14.671  0.800  1.050  0.000  300.0  11.375  0.0
.CHAIN VP24    14.566  0.800  1.050  0.000  300.0
.CHAIN SW17    14.503  0.800  1.050  0.000  300.0   2.126  0.0
.CHAIN VP26    13.791  0.800  1.050  0.000  300.0
.CHAIN SW19    13.796  0.800  1.050  0.000  300.0   1.870  0.0
.CHAIN VP27    13.403  0.800  1.050  0.000  300.0
.CHAIN SW20    13.327  0.800  1.050  0.000  300.0   1.821  0.0
.CHAIN VP28    12.582  0.800  1.050  0.000  300.0
.CHAIN SW21    12.527  0.800  1.050  0.000  300.0   1.852  0.0
.CHAIN VP29    11.679  0.800  1.050  0.000  300.0
.CHAIN SW23    11.588  0.800  1.050  0.000  300.0   1.875  0.0
.CHAIN VP30    10.511  0.800  1.050  0.000  300.0
.CHAIN SW25    10.450  0.800  1.050  0.000  300.0   1.875  0.0
.CHAIN VP31    9.390   0.800  1.050  0.000  300.0
.CHAIN SW26    9.213   0.800  1.050  0.000  300.0   1.648  0.0
.CHAIN VP33    12.771  0.800  1.050  0.000  300.0
.CHAIN SW31    12.718  0.800  1.050  0.000  300.0   2.125  0.0
.CHAIN VP34    11.286  0.800  1.050  0.000  300.0
.CHAIN VP5     9.824   0.800  1.050  0.000  300.0  85.327  0.0
.MANH  EX      9.140   0.800  1.050
.CHAIN  VP9    11.790  0.800  1.050  0.000  300.0
.CHAIN SW32    11.950  0.800  1.050  0.000  300.0  24.433  0.0
.CHAIN VP10    11.060  0.800  1.050  0.000  300.0  43.994  0.0
.CHAIN SW33    10.470  0.800  1.050  0.000  300.0  57.532  0.0
.CHAIN VP11    10.399  0.800  1.050  0.000  300.0  26.673  0.0
.CHAIN SW34    9.609   0.800  1.050  0.000  300.0  86.873  0.0
.CHAIN VP12    9.361   0.800  1.050  0.000  300.0  87.071  0.0
.CHAIN  EX     9.140   0.800  1.050  0.000  300.0  80.081  0.0
.END
```