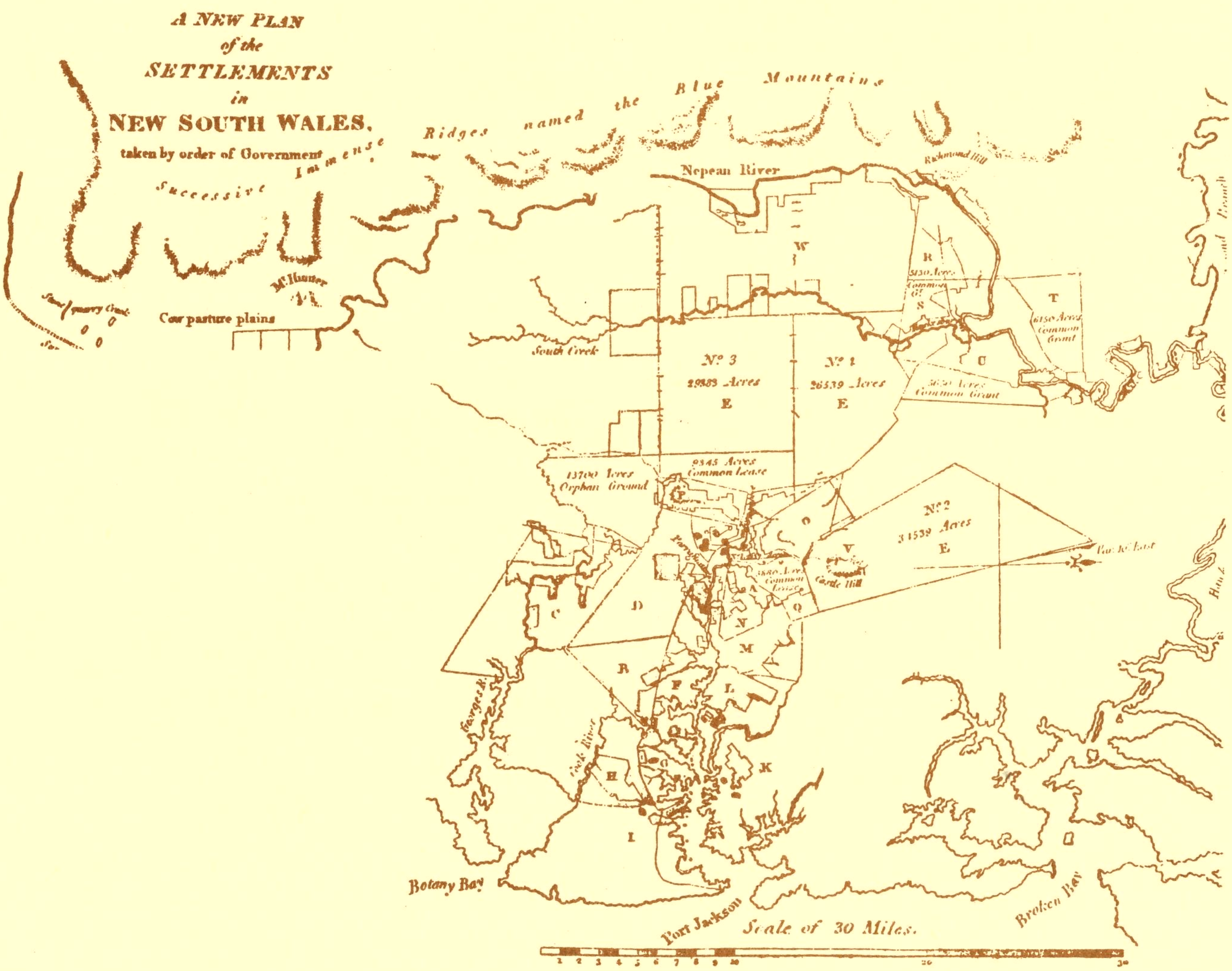
DATA UPDATE IN A LAND INFORMATION SYSTEM

ROBIN C. MULLIN



UNISURV S-34, 1988

Reports from

SCHOOL OF SURVEYING

SCIENTIA

MANU ET MENTE

THE UNIVERSITY OF NEW SOUTH WALES KENSINGTON, NSW AUSTRALIA 2033

UNISURV REPORT S - 34, 1988

DATA UPDATE
IN A
LAND INFORMATION NETWORK

Robin C. Mullin

Received: July, 1988

SCHOOL OF SURVEYING
UNIVERSITY OF NEW SOUTH WALES
P.O. BOX 1,
KENSINGTON, N.S.W. 2033
AUSTRALIA

ABSTRACT

This research deals with the problem of the on-going update of data exchanged in a land information network. In the past, major developments have been undertaken to enable the exchange of data between land information systems. Once a copy of a data set has been exchanged two questions must be faced: **What happens to a copy of land information, collected and maintained by another agency, once it updates its information? How are data updates propagated in a land information network?**

A model of a land information network and the data update process have been developed. Based on these, a functional description of the database and software to perform data updating is presented. A prototype of the data update process was implemented using the ARC/INFO geographic information system. This was used to test four approaches to data updating.

The four approaches are: bulk, block, incremental, and alert updates. A bulk update is performed by transferring a complete copy of an updated file and replacing the superseded file with the updated one. A block update requires that the data set be partitioned into blocks. When an update occurs, only the blocks which are affected need to be transferred. An incremental update approach records each feature which is added or deleted and transmits only the features needed to update the copy of the file. An alert is a marker indicating that an update has occurred. It can be placed in a file to warn a user that if he is active in an area containing markers, updated data is available.

The four approaches to data updating have been tested using a cadastral data set. They have been shown to provide a variety of solutions to the update problem; covering a wide range of capability and performance requirements. Due to the variability of the data updating environment, it is not possible to develop specific criteria for the selection of optimum update policies. General criteria, which can be used in an iterative process, to reduce the number of available options and thus enable detailed investigations of the few most suitable approaches are presented.

ACKNOWLEDGEMENTS

I wish to thank my supervisor Associate Professor John Trinder and acknowledge his patient assistance in the writing and editing of the thesis. Thanks are due, as well, to Mr. Peter Zwart, University of Tasmania for his valuable comments in the initial development of the topic.

The research has been carried out with the financial assistance of the Dean of Engineering, University of New South Wales and this is gratefully acknowledged.

Thanks to Dr. Ewan Masters for his comments and to Miss Pat Sin for providing the test data set. Special thanks to my wife, Susan, for her encouragement and long hours of assistance with the typing, editing, and layout.

Finally, I wish to express my appreciation to the individuals in government and private agencies throughout Australia who have given of their time and experience to assist me in performing this research.

Table of Contents

Table of Figures

Table of Tables

Chapter 1 INTRODUCTION

In the last two decades, we have become increasingly aware that our natural resources are not unlimited and we have experienced the increasing pressures of growing urbanization. Administrators, especially within the public sector, are being called upon to achieve more and more with less and less funding. There are pressures on all sides to maximize the utilization of our resources. As a result, managers are requesting timely and accurate information upon which to base decisions.

Agencies which deal with information related to land have been strongly affected by the changing times. There are ever growing masses of data and requirements for its use. This has led to the introduction of computerized land information systems. The distinctive characteristic of these systems is the ability to integrate, analyze, and rapidly access data via a spatial referencing system. This is normally achieved by linking map data, in a computer graphics system, and textual data, in a database management system.

Due to the strong nexus between human activity and the land, land information plays a powerful role in inter-relating the various functions of government and business. Thus, land information systems have found application in a wide variety of fields, including: legal, fiscal, natural resource, environmental, infrastructure, and socioeconomic management. There has been the proposal, development, implementation, and operation of a number of land information systems throughout the world. These have had varying degrees of success and have utilized a broad spectrum of technologies. One common factor appears to be emerging, namely, the requirement for data in the system to be shared with other systems, all of which are linked together to form a land information network.

1.1 The Data Update Problem

One of the cornerstones of the cost/benefit justification of land information systems has been the ability to reduce duplication of effort in the collection and maintenance of land related data by the sharing of data. To this end the land information community has struggled with the problems of data exchange between dissimilar hardware and software systems networked together. As well, users of land information systems acknowledge the dynamic nature of land related data and see great potential for enhanced decision making based on current, up-to-date data. The dynamic nature of land information confronts us with a major issue once the data exchange problem has been dealt with, namely: **What happens to a copy of the land information, collected and maintained by another agency, once it updates its information? How are data updates propagated in a land information network?** The answer to these two questions is the central focus of this research.

1.2 Objectives of the Research

This research had the following objectives:

1) Develop conceptual models of a land information network and the data update process in a land information network.

2) Develop alternative approaches to updating data in a land information network, the primary emphasis being on updating graphics data.

3) Formulate a means of implementing the data update process (data update server) and the alternative approaches to updating graphics data in a computerized land information network. The functional description of the implementation must be flexible enough to be incorporated into a wide range of land information systems utilizing differing hardware, software, and data structures.

4) Develop software to implement a prototype of the data update server and the alternative approaches to updating with the Environmental Systems Research Institute's ARC/INFO geographic information system on a Digital Equipment Corporation VAX/VMS computer system. The prototype is to incorporate facilities to operate in an automated fashion or under manual control.

5) Test the prototype system on an actual data set to determine the system's viability, performance, and resource requirements.

6) Based on the test results, establish specific criteria for the selection of optimum update policies for land information systems.

The objectives of the research have, for the most part, been fulfilled. It has been determined that due to the variability of the data updating environment it is not possible to develop specific criteria for the selection of optimum update policies. Instead, general criteria are presented.

1.3 Overview of the Thesis

Chapter 2 defines the terms land information system and land information network as they are used in this thesis. It also describes common configurations of land information networks and the dominant issues involved in integrating land information systems into a cohesive and viable network. This is intended as an introduction to the integration issues and is not an exhaustive treatment of the subject.

Chapter 3 outlines the impetus for data updating in a land information network. It briefly discusses the need for data update in a land information system and the potential advantages of data exchange in a land information network. The conclusion that data which is exchanged must also be updated is drawn and supporting comments from land information network developers are presented.

Chapter 4 presents the conceptual basis for the research. It begins with a model of a land information network and uses this to develop a model of the data update process. On the basis of these models it is concluded that facilities to record and track data updates are required. It is proposed that there are four approaches to data updating: 1) bulk, 2) block, 3) incremental, and 4) alerts.

Chapter 5 contains a functional description of the facilities to record and track updates and the four approaches to updating. It is intended that this description provide the basis for the implementation of the data update model in a land information system/network.

Chapter 6 describes how the functional description of the data update process, presented in Chapter 5, was prototyped using the ARC/INFO geographic information system software. It includes descriptions of the relations used in the database management system and pseudo-code for the primary data update software which has been developed. As well, the test data set is described and the results of testing presented.

Chapter 7 draws together the requirements of the functional description in Chapter 5 and the results of the testing in Chapter 6 to develop criteria for the selection of an optimum update policy. The selection is based on capability, performance, and cost/benefit criteria. As well, a systematic approach to selecting an optimum update policy is suggested.

Chapter 8 contains conclusions, recommendations, and suggested areas of further research arising from this investigation of data updating in a land information network.

The issue of the on-going update of data exchanged in a land information network is one which is raised only infrequently. On these occasions it is unanimously declared to be of vital importance and is then tabled for future consideration while more immediate and pressing problems are dealt with. It is hoped that this research will stimulate interest in, not only raising but, addressing the "immediate" problem of data update in a land information network. It is also hoped that the research, presented herein, will contribute to the understanding and solution of this problem.

Chapter 2 THE LAND INFORMATION NETWORK CONCEPT

The process of recording information about the extent and characteristics of land is as old as land itself, but in the past twenty years there has been increased concern over the efficient storage, retrieval, and analysis of land related data. This has led to the utilization of computer technology to assist in the handling of large and complex land related data sets.

These computer systems have come to be known as geographic or land information systems. It is not intended to become embroiled in the controversy surrounding the taxonomy of these systems [HAMILTON et al., 1984, MARBLE, 1984] and thus for the purpose of this research the International Federation of Surveyors' (FIG) definition of a land information system will be adopted.

> A **Land Information System** is a tool for legal, administrative and economic decision-making and an aid for planning and development which consists on the one hand of a data base containing spatially referenced land-related data for a defined area, and on the other hand, of procedures and techniques for the systematic collection, updating, processing and distribution of the data. The base of a land information system is a uniform spatial referencing system for the data in the system, which also facilitates the linking of data within the system with other land-related data. [1981]

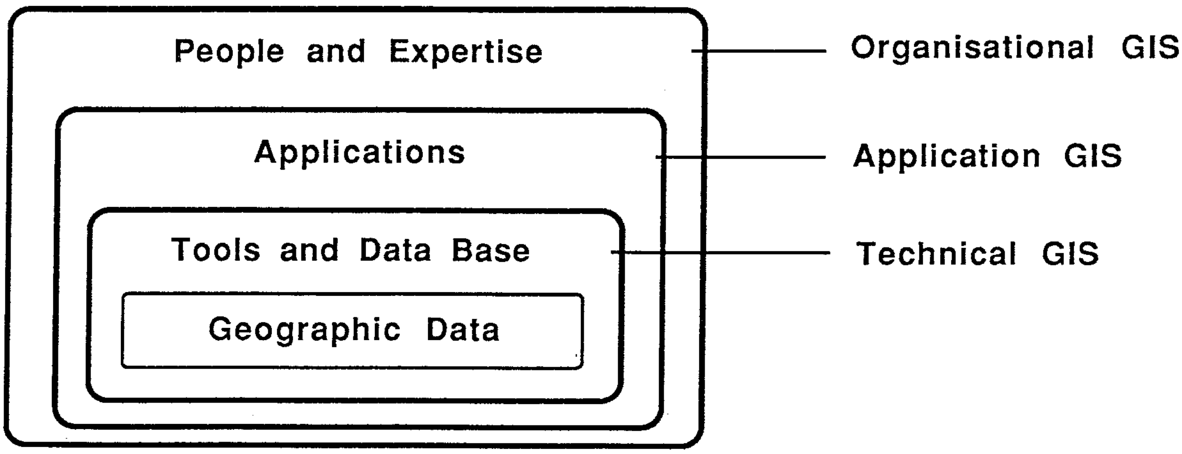


Figure 2.1 **Interaction of Various Aspects of a Geographic Information System** [MOREHOUSE, 1987]

The separation of organizational, application, and technical components of these systems is more clearly defined by Morehouse [1987]. His concept of the interaction of these components in a geographic information system is depicted in Figure 2.1.

The land/geographic information systems market is growing rapidly. One forecast predicts that $200 billion per year will be spent on land information systems by the year 2000 [PRICE, 1987]. The dramatic interest being shown in land information systems is due to the benefits which can result from a properly designed and implemented system [DAW, 1987]. The United Nations Economic and Social Council [1973] stated that:

> "an efficient land records system is a basic tool for stimulating economic and social development in both rural and urban areas, and for ensuring effective administration and planning in the public sector."

One of the central benefits of a land information system is the savings resulting from a reduction in the duplication of effort in collecting and maintaining land related data. Agencies involved in land information systems have seen the benefits within their organization and now are looking to realize the same economies, in a wider sphere, by sharing data between land information systems. This trend towards the linking of land information systems has led to the concept of a land information network.

Palmer [1984, pp.44] has defined a **land information network** as:

> "a confederation of individual member systems. While a land information system may be regarded as an attempt to improve the effective flow of information within an organization, a network may be viewed as an attempt to improve the effective flow of information between organizations."

Major initiatives are currently under way in Australia to develop state-wide land information systems [BELL, 1987, EDDINGTON, 1987, HART, 1987, HYDE, 1987, PHILLIPS et al., 1987, ROBERTS, 1987, SEDUNARY, 1987, STEPHENS, 1987]. These would be better described as land information networks, as they seek to link systems containing legal, fiscal, environmental, and socioeconomic data and to utilize spatial references to integrate these diverse data sets. Similar developments in land information networks are under way overseas [GAUDET et al., 1984, KENNEDY, 1986].

2.1 Land Information Network Configurations

Land information networks can be formed in a variety of configurations. In general, their organization is an adaptation of digital communication network topologies, although this does not necessarily mean that they utilize a digital communication network. The three most common configurations are: 1) fully connected, 2) hub, and 3) nodal. Figure 2.2 depicts the linkages between state government departments in New South Wales regarding land parcel title details which existed in 1983 [INTER-DEPARTMENTAL COMMITTEE ON LAND INFORMATION (NEW SOUTH WALES), 1983]. This is an existing, for the most part non-computerized, land information network which exemplifies a fully connected configuration. Figure 2.3 shows the proposed configuration of the New South Wales state-wide land information system (network) as a central hub which acts as a indexing and switching mechanism [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987a]. Figure 2.4 illustrates the South Australian nodal approach to land information networks with nodes containing functional databases which are interconnected and connected to peripheral users and databases [SEDUNARY, 1984].

It is clear that the fully connected configuration can lead to extensive and uncontrolled duplication of effort and data. This reflects the current situation for most conventional land information networks. The hub configuration provides a secure environment and removes the bulk of the burden for data dissemination from the data trustees. Its opponents consider it to be a monolithic centralist system with too much data and power concentrated in the hub. The nodal approach allows users to access data directly from its source. Opponents of the nodal approach consider it to place too heavy a burden on data trustees for the dissemination of data, the routing of requests for data to the appropriate trustee to be overly complex, and the network to be

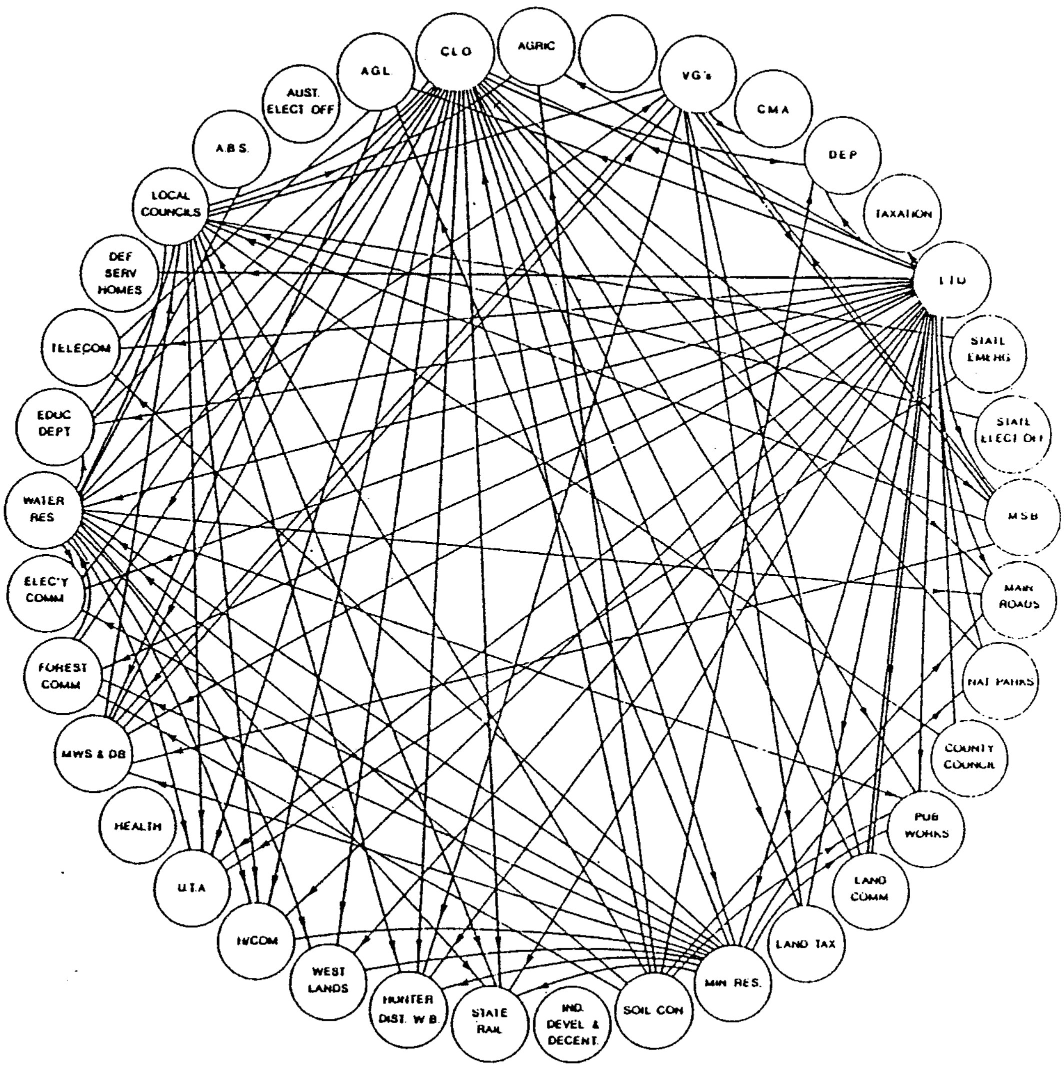


Figure 2.2 **Example of a Fully Connected Configuration Land Information Network** [INTER-DEPARTMENTAL COMMITTEE,1983]

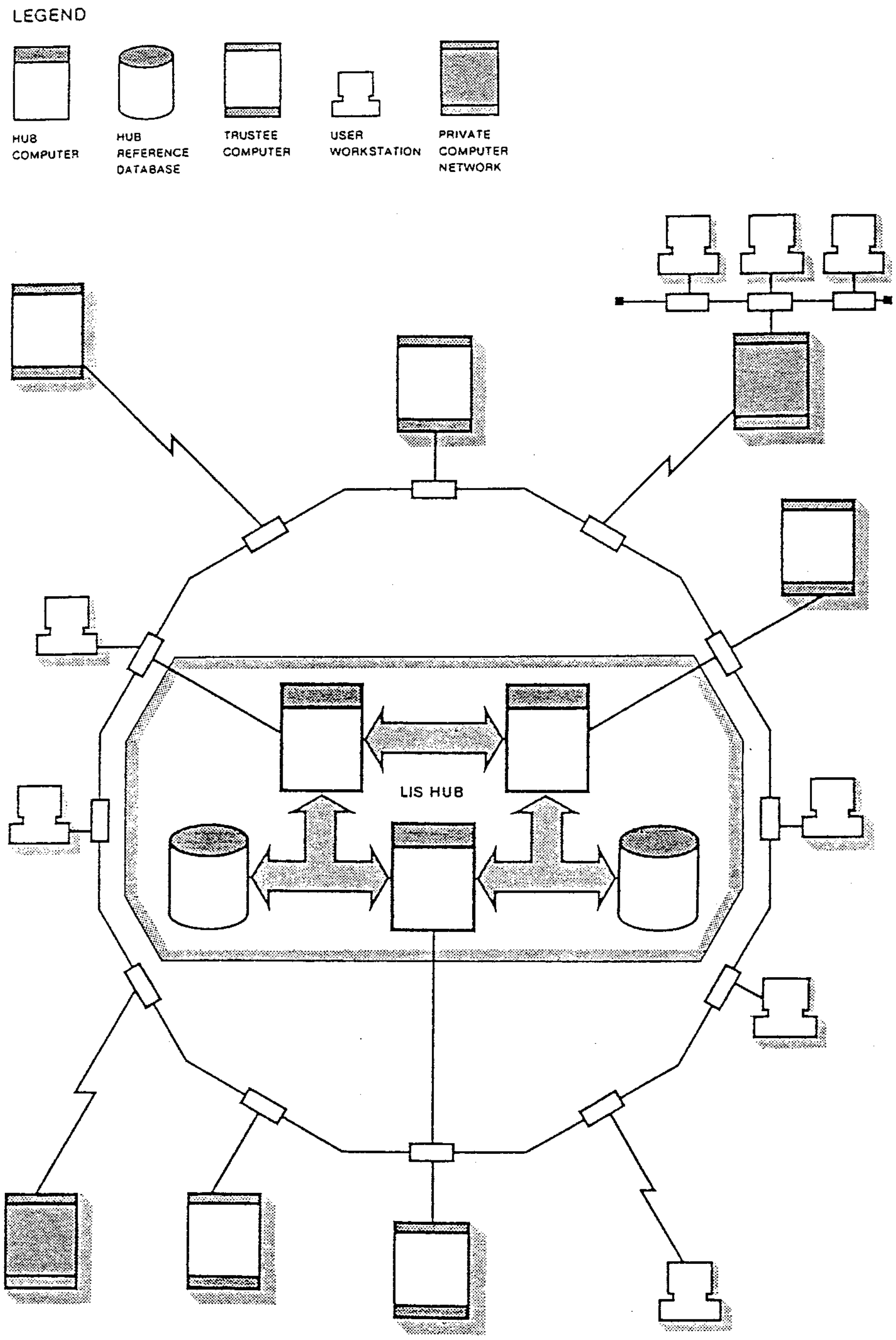


Figure 2.3 **Example of a Hub Configuration Land Information Network** [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES),1987A]

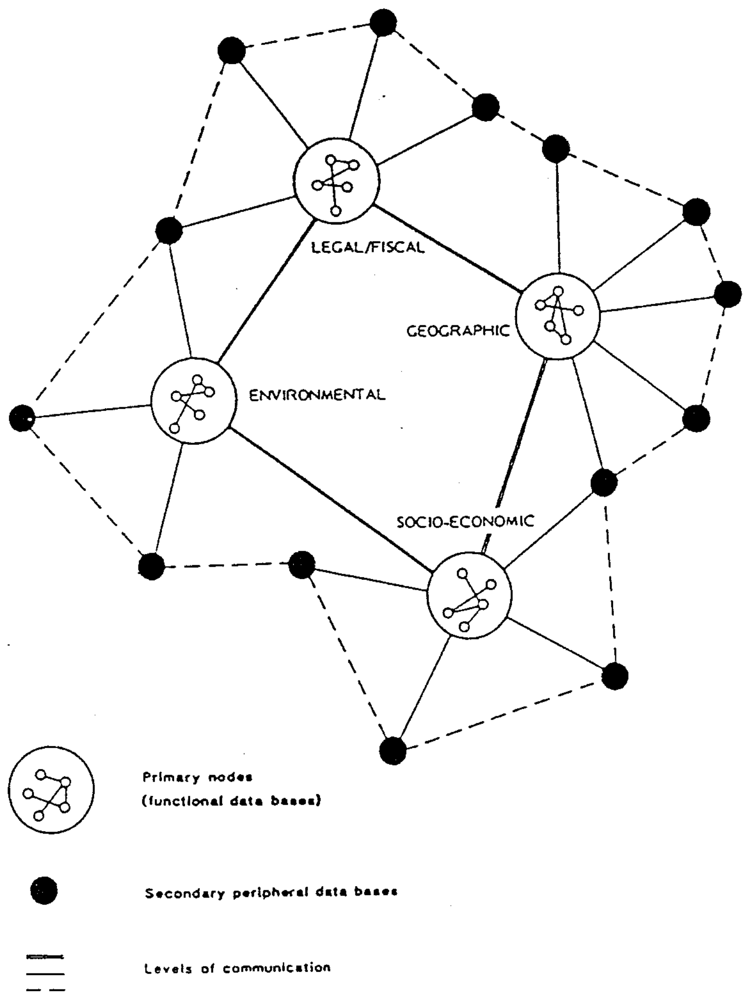


Figure 2.4 **Example of a Nodal Configuration Land Information Network** [SEDUNARY,1987]

vulnerable to security breach. For the purposes of this research it is not important which configuration the land information network utilizes. Section 4.1 describes the model of a land information network upon which this research is based.

Within a land information network there are essentially three types of nodes: 1) storage, 2) inquiry, 3) analysis. This classification is an adaptation of the classifications given by Hebblethwaite et al. [1986] and Pullar [1985]. In reality, any given node in the network may be a combination of these types and function differently depending upon the data set being acted upon. A storage node is one where data is collected, verified, and stored. It also services requests for data from other users in the network. An inquiry node stores no data of its own but merely queries other databases in the land information network. An analysis node obtains a copy of a data set from a storage node and performs analysis on the data set. The storage node is responsible for updating the data set and will always have the most current information available. An inquiry node stores no data of its own and thus any update which occurs on the data set it is querying is immediately reflected in the answers to its queries. Since the analysis node contains a copy of a data set, any updates which occur on the original copy of the data set must be propagated to the analysis node and applied to that data set to bring it up-to-date. This problem of maintaining the currency of copies of data sets in a land information network is the central issue of this research.

2.2 Integration Issues

The fundamental problem in a land information network is the integration of the land information systems which comprise the network. This is not an easy task. Land information systems may be at widely varying levels of technical sophistication and be heterogeneous with respect to computer hardware, software, data structures, and communication facilities. Essentially there are three main integration issues:

1) administration;
2) data directories;
3) standards.

Solutions to the problems presented by these three areas must be resolved before a land information network can function properly. This is independent of the technology used to establish the network.

The following sections are not intended to be definitive statements on these issues of system integration but serve to introduce and cultivate an appreciation for the problems involved.

2.2.1 Administration

The administration of land information networks is a task which is far more difficult than any of the technical problems which are currently being faced. It requires a shift from the traditional view of information in government and the marshalling of support from a wide array of disciplines and agencies, in both the public and private sectors. Humphries [1984, pp.11] addresses these issues in saying that:

> "The "traditional" role of government is to assign a statutory responsibility to a particular department. The department generates information which is then assumed to be departmentally "owned". The truth of the matter is that government, as the body corporate, has the ownership of information and should not be beholden to the "tunnel vision" of individuals or individual departments. The absence of a corporate infrastructure, however, leaves departmental management with little option but to foster exclusive self interest.
>
> In order to respond to the global problems of the management of land information, government needs to take a corporate view of the problem. It needs to identify original sources, rather than secondary sources, of information and to build the infrastructure of a land information system to act as a concentrator and disseminator of that information to others."

In order to deal with these issues effectively, it must be acknowledged that the support of the political masters of government must be obtained and retained. Justice Kirby [1982, pp.16] addresses this point by saying:

> ".....it is absolutely vital that elected officers of government address the complex institutional problems that exist. Without a commitment by the Executive Government, vested departmental interests will undoubtedly proclude rationalisation of land management systems. The problem is not to be solved, I believe, by the simple expedient of assigning the co-ordinating role to a land related department. Such departments are able to address the functional needs of a system. But of equal importance is the need for financial co-ordination (involving the Treasury), organisational co-ordination (involving the Public Service Board) and co-ordination of department politics (involving, normally, the Premier's officers)."

It must also be recognized that not all information is exchanged as "hard" data. Palmer [1984] categorizes information as explicit and implicit. Explicit information is comprised of facts and data which are usually stored in land information systems. On the other hand,

> "Implicit information may result from intuition, speculation or supposition and may be based on years of experience. It includes unquantifiable ideas and subtle relationships between the occurrences of events and problems under consideration. This implicit information may be vital to the effectiveness of an organization. To ignore it is to avoid a fundamental problem."[PALMER, 1984, pp.39]

Thus a land information network is required to provide mechanisms for the interaction of decision makers and others, who capture and utilize the information in the land information network, on a personal and informal level. This highlights the importance of special interest groups and user forums in a land information network. These groups and forums allow and encourage the flow of implicit information between individuals.

2.2.2 Data Directories

The role of the data directory facility in the Alberta Land Related Information System Network is outlined by Kennedy [1986, pp.442]:

> "A fundamental user requirement of the LRIS *(Land Related Information System)* Network is a mechanism to identify what data is available, the quantity and quality of that data, who has the data, and how the data may be accessed."

The benefits of such directories have been reported by the Australian Commonwealth Land Information Support Group [1987, pp.6] based on their experience with the "Landsearch" directory:

> "For the user, a number of advantages derive from readily acquired knowledge of available data sets:
>
> - awareness of the vast pool of data already available to meet information needs.
> - identification of duplication, gaps and needs in data collection.
> - identification of areas of expertise.
> - provision for easier information flows within and between government.
> - timely access by government to its own information.
> - facilitation of research.
> - greater public understanding of government activity and decision making."

The benefits of these directories are unquestionable but they require an extensive commitment of resources to create and an on going commitment to maintain. The problems of indexing widely varying data sets with a standard format, providing facilities to query the data spatially, attaching quality indicators, and determining the accessibility to sensitive data sets must be overcome. The provision of on-line systems which service requests for information by determining the location of data items in the data directory and then sending a request for the information over a digital communications network adds an order of magnitude to the complexity of the issue.

2.2.3 Standards

For data and updates to be exchanged in a land information network, standards must be agreed upon in the following areas:

1) **Terms and Definitions:** It is important to realise that data must be interpreted before it becomes information. We interpret data based on our past experiences and training. Thus, a reference to a "parcel" of land may have very different meanings for a land surveyor and a property assessor [BEDARD, 1986]. In order to exchange data meaningfully, we must ensure that the sender and receiver share a "common experience" or that we explicitly define what is meant by the data items which are being exchanged. The importance of standard terms and definitions is highlighted in the U.S. National Committee on Digital Cartographic Data Standards draft standard [MOELLERING, 1987].

2) **Data exchange formats:** Standard data exchange formats are required to transfer data between computerized land information systems with differing structures for the organization and storage of data. There are currently, at least, fourteen formalized standard formats for the exchange of graphical land information [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987d]. As yet, no single format has emerged which has gained wide acceptance. Existing land information networks utilize either mutually agreed upon customized formats or vendor specific interchange formats, e.g. Intergraph's SIF format.

3) **Communication systems:** The underlying foundation of any land information network is the communication system which links the land information systems and passes data between them. Land information networks, which require facilities to transfer data on-line, need to establish standards for communication between systems. In environments where all of the land information systems in the network utilize equipment from a single hardware vendor, the solution is usually provided by using the vendor's computer networking product. Unfortunately, this is rarely the case. Vendors have developed products which will allow the interconnection of various proprietary networks but this involves

specialized hardware, software, and procedures. As with data exchange formats, a wide variety of standards exist for communications [FEELEY, 1986]. High expectations are being placed upon the International Standards Organization's Open System Interconnect (OSI) standards. This set of standards seeks to define the parameters necessary for transparent application program to application program communication across computer networks. Developments in this area will prove to be a significant boon to the development of land information networks.

The development of standards is a long and exacting process but the potential savings are well worth the effort. The words of Justice Kirby [1982, pp.14] could well speak to the need for standards upon which to establish land information networks:

> "The technology does not stand in the way. Only our local obsessions, a lack of national vision and puny, parochial attitudes, limit the development of the common standards and definitions necessary to establish a land use data bank for Australia."

2.3 Summary

In summary, the concept of a land information network is a natural extension of land information systems development. It seeks to realize the same benefits associated with land information systems on a broader scale by linking land information systems into a network.

The success of a land information network will depend upon the administrative framework established to facilitate the effective interchange of information, the data directories established to identify sources and characteristics of land information, and the standards adopted by the network to allow the ready interchange of information in a heterogeneous environment.

Chapter 3 REQUIREMENT FOR DATA UPDATE

The following section addresses the requirement for up-to-date data in a land information network. The requirement for up-to-date data in a land information system is examined first, then the need to exchange data in a land information network is raised, and finally the need for updates to the exchanged data is presented.

It is universally accepted that land information systems should provide readily accessible and current information to their users. Thus, in the initial stages of system planning, design, and cost justification the role of data updating figures prominently in the discussions. As well, in the development of land information networks the idea of various users obtaining copies of existing data sets without incurring the cost of data collection and maintenance is high on the agenda. The marriage of the idea that data is dynamic and must be updated, with the idea of exchanging data does not usually occur, and only passing mention is made of the need to continue to update data which has been exchanged in the land information network.

The ad hoc development that often characterizes the creation of land information networks neglects or ignores the issue of data updating. Even in developments where a formal plan is defined, the initial tasks of creating large databases and creating data interchange mechanisms relegates the development of facilities to update data which has been exchanged to some later stage of the system implementation. This later stage of development will become a priority when agencies with copies of data sets decide that it is less costly to maintain the data themselves than to again perform the entire data exchange from the data trustee to capture a, possibly small, number of updates. Thus it is of paramount importance that land information systems incorporate, as early as possible, facilities to record and transfer updates to data exchanged with other land information systems. As well, any negotiations between land information systems managers for obtaining copies of digital data should include discussions of how subsequent updates will be handled. If this does not occur, the land information network runs the risk of reverting to a situation of redundant systems collecting and maintaining redundant data.

Beyond developing mechanisms in the land information network to provide updates to exchanged data sets, the land information system providing the updates must comply with a minimum standard for the production of timely updates. This minimum standard must be agreed to by all of the users of the data in the network. If this standard of quality is not achieved the users will lose confidence in the fidelity of the data and be uncertain as to its fitness for use. Bedard [1986] states that uncertainty in a land information system can be dealt with in two ways: 1) reduction, or 2) absorption. By providing up-to-date data, the uncertainty that the data in a land information system does not represent reality at this time is reduced. The absorption of uncertainty occurs when the provider of the information:

> "...guarantees his model of the reality and compensates the users who get damages due to erroneous data from the guaranteed model. Uncertainty absorption also takes place when a user of data utilizes non-guaranteed models, then it is not the provider of data who absorbs the uncertainty but the user. ... The level to which a user is willing to absorb the uncertainty determines his confidence level in the LIS. As we know the confidence in a system is a key element for its use..." [BEDARD, 1986, pp.261-262]

The concept of a data provider guaranteeing the correctness of data is not new. It is a cornerstone of the Torrens land titling system and well established in government produced hydrographic charts. The Torrens titling system is a simplified example of the need for updating since the register of title is centralized. The government guarantees the accuracy of the information contained on the title document and will reimburse any user who incurs damages resulting from acting upon erroneous information.

The difficulties in updating hydrographic charts more closely approximate the problems which are presented by a land information network. There may be many copies of a hydrographic chart and as changes occur they are distributed in the "Notice to Mariners" bulletins. The mariner is then responsible to update his chart accordingly. The hydrographic service is legally liable for any damages which occur due to faulty information on hydrographic charts but, as well, the mariner is required by law to carry an up-to-date copy of the hydrographic charts for the area in which he is navigating. The hydrographic service is the only official source of charts and updates. Thus, the legal framework enforces adherence to the hydrographic service's updating mechanism.

The provision of timely data updates serves to instill confidence in the system which results in its increased utilization. It is preferable for the users to absorb residual uncertainty in the system than to resort to the legally declared solution.

The level of uncertainty which a user can tolerate will depend upon the application involved. For example:

> "In Australia, it has been estimated that some $200 million are wasted annually on repairing utility services which are damaged during construction work." [SEDUNARY, 1986, pp.5]

Obviously, the provision of current and accurate utilities information could result in substantial savings to the utility companies and subsequently rate payers. Blachut et al. [1979, pp.351] make strong statements on the requirement for current data in urban land information systems:

> "The alternative to continuous maintenance of city survey data and maps is wasteful and expensive duplication, lack of foundation for a rational administration and planning, and general chaos."

It is important to note the use of the word "continuous". "Continuous" can be defined as "without interruption". This should not be confused with "instantaneous". Data updating in a land information network must be seen as an on-going process but not necessarily instantaneous [GRSKOVICH, 1978]. The greater the degree of timeliness required, the greater the cost of the system.

The requirement for up-to-date data in a land information network is highlighted in "The Conceptual and Technological Framework for the New South Wales Land Information System" [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987b, pp.43] when it is stated that:

> "The LIS *(LIN)* database, effectively, will be a distributed database with the non-redundant source data kept by data trustees and redundant copies of the source data kept by many data users. Maintaining consistency and currency of these distributed copies of data is one of the most important factors in the implementation of the LIS network concept."

Even with the acknowledgement of the importance of the data updating problem, operational land information systems and networks have not addressed this issue. For example, the world leading South Australia land information network has yet to come to grips with the issue:

> "While the Interim L.I.S. Operational Policy does lay down criteria that attempt to ensure that land related data bases can be interrelated, it does not address the more practical aspects of providing, **updating**, integrating and disseminating such data." [SOUTH AUSTRALIAN DATA PROCESSING BOARD, 1985]

It is a popular misconception that the data update problem can be resolved by eliminating all duplicate copies of land information and providing one up-to-date database for everyone's use. The sheer volume and complexity of land information and the diversity of users of land information preclude this centralist approach. MacIver [1984, pp.247] states that:

> "Seeking greater economy through centralization of disparate types of land records often results in achieving a false economy. A better way to approach the issue of reducing overall data related costs is to seek an optimum data redundancy through use of a coherent data exchange methodology. If data sharing is facilitated, data redundancy will tend to eliminate itself without the need for centralized decisionmaking and complicated consensus building."

Extensive research [PULLAR, 1985, HEBBLETHWAITE et al., 1986, STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987] has demonstrated that land information must be duplicated amongst users in a network to provide the required ease of accessibility and analysis. Goh [1987, pp.5] when addressing this issue of minimization of redundancy states that:

> "Physical duplication of data is in itself not a major concern, especially when computerized storage costs are declining rapidly. Even in distributed processing, some designs may necessitate duplicate databases at the distributed nodes. **The main concern of any organization is to minimize overheads involved in collecting and maintaining the data and to institute appropriate mechanisms to ensure timely updates to the various data bases."**

In summary, it is well recognized that the data in a land information system or network must be up-to-date if it is to be relied upon by the users of the system. As well, the provision of policies and procedures for data updating are considered of central importance for the on-going success of land information systems and networks. Yet, the pressures of creating databases for land related data have pushed the issue of data updating into the background. Unfortunately, the exchange of land related data often occurs without the issue of subsequent data update being addressed. This research is an attempt to redress the deficiencies in current thinking on data updating. In conclusion, it can be said that:

> **"The issue of requirements for maintaining up-to-date land-related data records is critical for the success of the whole LIS. The questions: how, and when the land-related data records will be updated must be answered before the implementation, for the LIS users to be able to use it properly and with confidence."** [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987b, pp.43]

Chapter 4 MODEL OF DATA UPDATE IN A LIN

The following sections develop a conceptual model for the data update process in a land information network. As with any model, it is a representation of reality and not reality itself. The model contains appropriate simplifications which highlight areas which impact on the data update process and hopefully suppresses irrelevant factors [PEUQUET, 1984]. The model serves as an initial test-bed for approaches to the update problem. If an approach will satisfy the requirements of the model, and the model is an appropriate representation of reality, then the approach has a reasonable chance of success.

Within the data update model there is no explicit requirement that the land information network be computerized. Each component or process in the model can be accommodated within a variety of levels of technical sophistication. This is in keeping with the realities of the current situation - land information systems may be at any and every level of technical sophistication. The approach which has been taken assumes that the structures and processes appropriate for an "automated" data update process in a computerized land information network can, with simplification and adaptation, be applied to less sophisticated levels of technology. In applying the data update model an attempt has been made to present options for a variety of levels of technology. The criteria for an optimum update policy takes the sophistication of the land information systems involved (Section 7.1) into consideration.

4.1 Model of a Land Information Network

For the purpose of modelling, the land information network has been viewed as being two inter-connected land information systems. One system is designated as the source system and the other the destination system as shown in Figure 4.1.

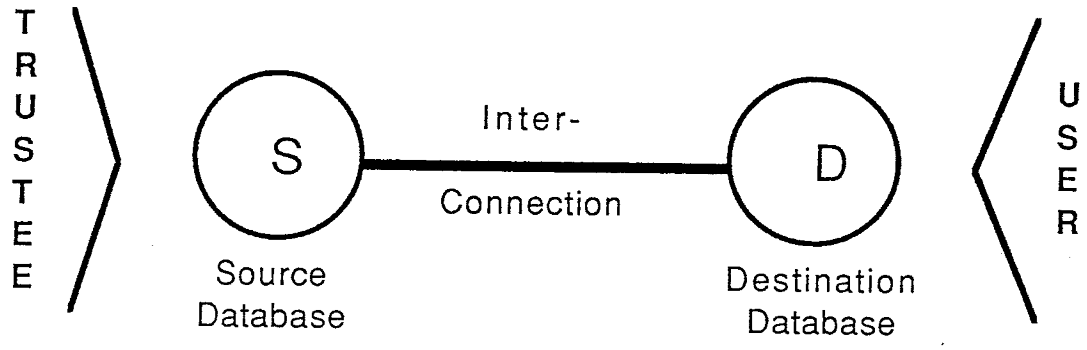


Figure 4.1 **Simplified Model of a Land Information Network**

The source is any land information system which provides a copy of a data set or sets to a destination system. The source system could be a data trustee, which collects and maintains a particular data set, or a land information bureau, with the responsibility of disseminating data. Both of these concepts, the trustee and the bureau (hub concept), exist in the New South Wales' state-wide land information system approach [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987a]. In other jurisdictions, e.g. Western Australia, South Australia, and Queensland, the concept of data trustees figures prominently.

The destination is any land information system which requires a copy of a data set or sets from a source system. The destination system must be a distinct entity from the source and maintain its own database. If the destination system is merely a satellite user of the source system's database, then no data update problem will occur. As updates are applied to the source system, the current information will be reflected in the satellite system's queries. It may appear to be folly to allow duplicate copies of data to pass from the source to a destination system, but as is described in Chapter 3, the goal of a land information network is the reduction of duplication of effort in the collection and maintenance of data, not necessarily the reduction of duplication of data. This means that the duplication of data must be controlled so that proper update procedures may be applied.

An example of a destination system would be any system which receives a copy of a data set from a data trustee, e.g. a mineral resources department receiving topographic data from a state mapping authority. Within the New South Wales' state-wide land information system approach, the hub system will be the destination system when receiving data from the data trustee. Once the hub has an authorized copy of the trustee's data set it will act as the source system to disseminate data to other destination systems, e.g. local governments (Figure 4.2). Thus, depending on the situation, any system in a land information network may operate as a source or destination system.

It must be assumed for generality that the source and destination systems are heterogeneous. That is, the two systems differ in the their operating hardware and software. The update process would be much simpler if the systems were homogeneous but this does not reflect the reality of the current situation. The approaches developed for the heterogeneous environment will be equally applicable for the homogeneous one.

To achieve transfers, there is clearly a requirement that the two systems be inter-connected. As will be shown, this stipulation is loose; with inter-connection being defined as a range of facilities from hardcopy transfer to magnetic media transfer to on-line digital communications.

The approaches to data update, developed in this model, can be applied to real world systems by decomposing an existing land information network into primitives which are pairs of inter-connected systems (Figure 4.1), one designated as a source and the other designated as a destination in each case. From these primitives, the full update process can be developed for a network of multiple systems in which the role of each system may alternate between of source and destination, depending upon the particular data set.

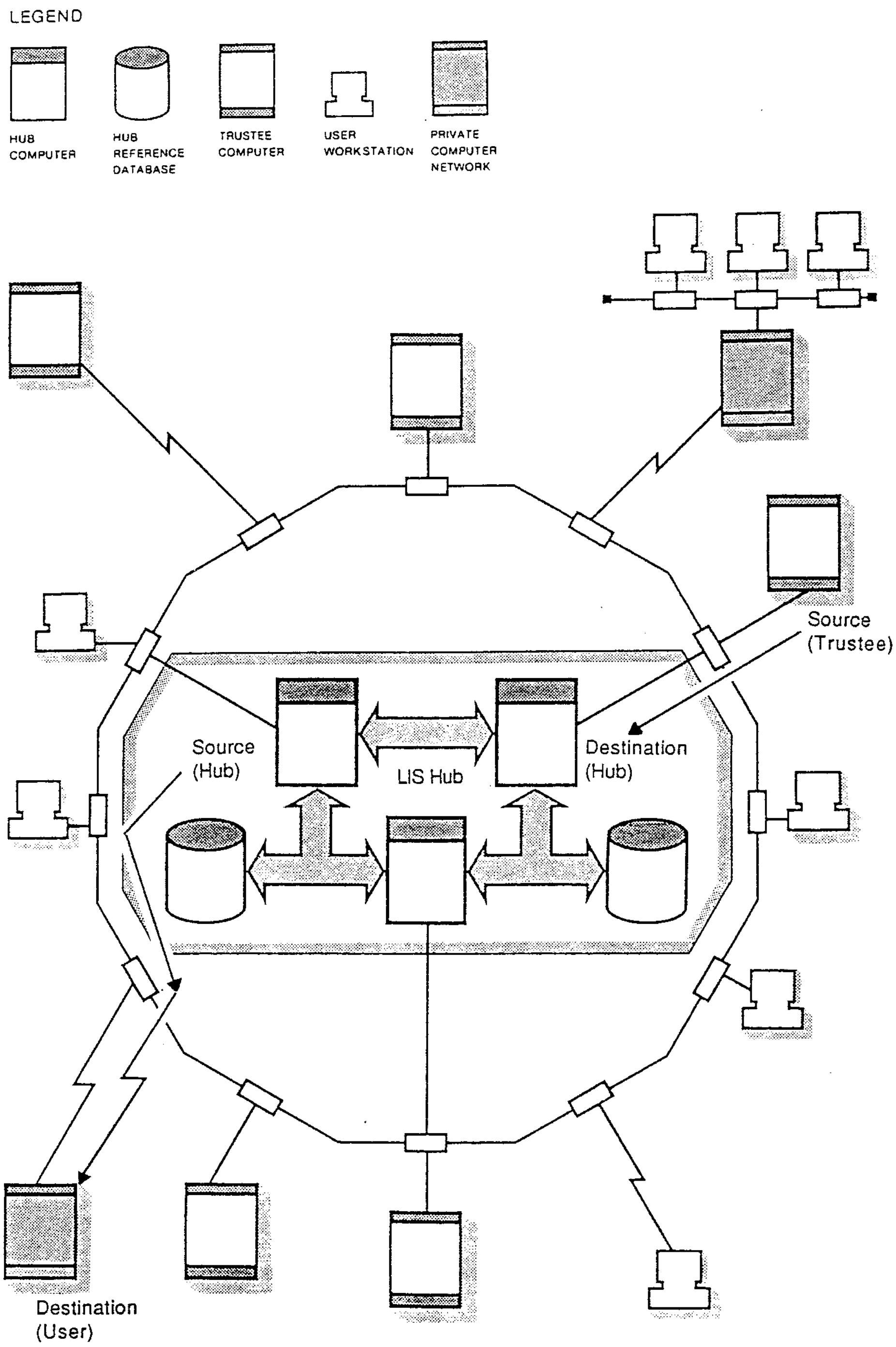


Figure 4.2 **Example of a Hub Acting as Both a Source and a Destination System** [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987a - modified]

4.2 The Update Facility

There are two foundational facilities which must exist before any update process can operate. First, the source database must accommodate a facility to record occurrences and location of updates performed by the trustee of the source database. If this does not exist, then it is impossible to detect whether an update has occurred or, if an update has occurred - it cannot be ascertained where it has occurred. Second, a database must exist which tracks when data sets are updated, a user's requests for notification of updates, and information on the subsequent fulfilment of update requests (Figure 4.3).

This highlights an ambiguity which exists in the use of the term "update". Within the data update model two types of updates exist. First, the source data set is updated directly by the data trustee. Second, the destination data set, which is a copy of the source data set, must have any updates to the source data set applied to it to ensure its currency. In the following discussion, the first type of update will be referred to as a "source update" and the second type will be referred to as a "propagation update"; since source updates must be propagated to the destination system for application to the destination data set.

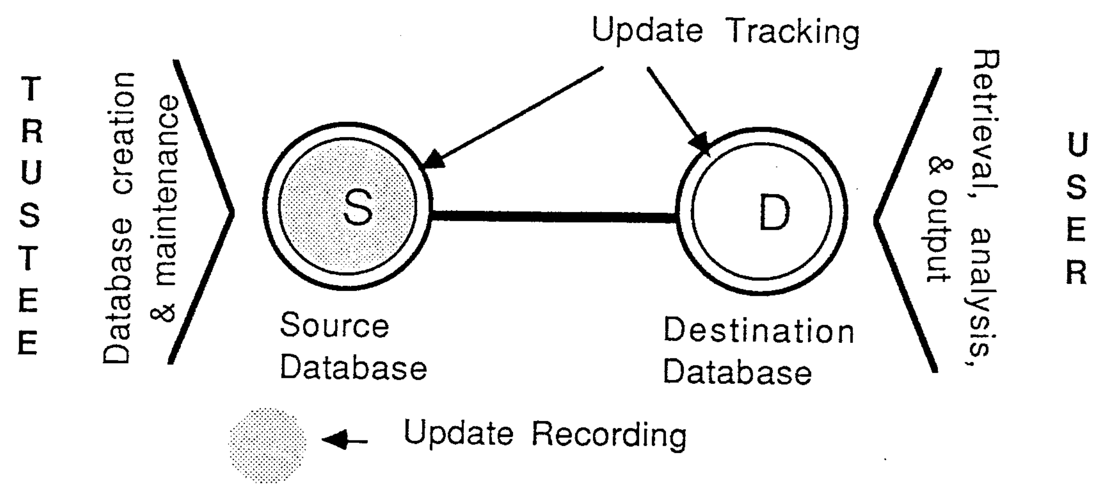


Figure 4.3 **Facilities to Record and Track Updates**

4.2.1 Facility to Record Updates

The objective of the facility to record updates is to systematically record the changes which occur in a data set such that any changes, which have occurred over a particular time interval, can subsequently be recalled and applied to a copy of the data set to update it. The structure to record updates deals with the issue of source updates. This research is not as concerned with the technology and methodologies of collecting data for the source update as with the detection and marking of these updates for future propagation to destination land information systems.

The facility to record updates acts as a time-stamping mechanism for the data in the land information system when updates are applied to the database. As can be seen in Figure 4.3, the recording of updates on the destination database is optional but will be present in most cases since most land information systems will act as both source and destination systems at some time. The level of resolution of the time-stamping will be determined by the level of sophistication of the land information system. It could consist of a record of when a particular file or area was last updated. A finer resolution would be the tagging of groups of features or records with the time of a change and the finest level of resolution would involve storage of additional attributes associated with each entity. These attributes would record when the entity is entered into the database and when it is superseded by more current data.

It is important to note that in some approaches to data updating, the recording of source updates not only involves the requirement that the data set is up-to-date at any time, but that the history of the changes must also be preserved. When deletions occur they may need to be archived so that they can be applied to an earlier copy of a data set, along with the additions in chronological order, to bring that copy up-to-date. As well, there may be a requirement to depict the data at different epochs for time series analysis.

4.2.2 Facility to Track Updates

The objective of the facility to track updates is to control the duplication of data and ensure that users with copies of data, have that data maintained to the level of currency appropriate for their application. The facility to track updates deals with both source and propagation updates.

The facility to track updates can be seen as a tracking database within each land information system containing information on the data sets within the land information system, their source, users who have copies of the system's data sets, the requirements for the currency of the data sets, and any outstanding requests for updates to these data sets. The files within this database are designed to be symmetrical, to accommodate the role of the land information system as a source system and/or a destination system in the land information network.

This tracking database serves as the control structure for the update process. When updates occur in the source data they are noted in this database and when propagation updates, for copies of the source data, are requested by users they are registered here. The update propagation process is initiated based on information in the tracking database and as the process progresses its status is recorded in the database.

As with the facility for recording updates, the facility to track updates may be implemented in a variety of ways depending upon the technical sophistication of the land information system involved. It may be a card file system or an on-line relational database. The important issue is the data, which must be collected and maintained to service the update process.

4.3 The Update Process

The update process involves both source and propagation updates. The source updates are detected and recorded as they occur while the propagation updates, which make up the largest portion of the update process, must be generated and tracked.

The update process model can be best described with the aid of diagrams, each depicting an operation within the update process, and the combination of these will result in the full update process model which serves as the basis for this research.

Figures 4.1 and 4.3 depict the land information network as two inter-connected land information systems, a source and a destination system, each with an update tracking database and, at least, the source land information system must have a facility for the recording of source updates. As well, the roles of the trustee and the user are noted.

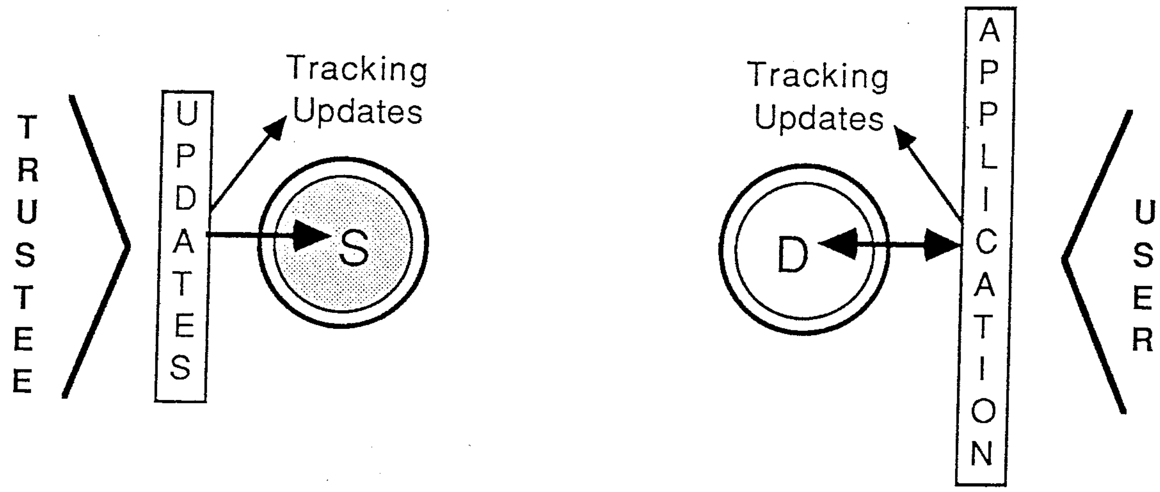


Figure 4.4 **Interaction Between Trustee/User and Source/Destination Databases**

In Figure 4.4, the interaction between the trustee and the source database, as well as, the interaction between the user and the destination database is shown. The trustee updates the source database. This is noted by both the facility for recording updates and the facility for tracking updates. The user accesses the destination database via application software. As well, the application may query the facility for tracking updates to check if the destination data set meets the currency requirements for the application.

When an application queries the update tracking database as to its currency, the update process must determine the source of the data and request from the source land information system's update tracking database whether an update to the destination database is required. As is seen in Figure 4.5, a negative response may be given to the request. This may be because no update is required or because the user does not have the privilege to access that particular data set. The process returns this information to the destination update tracking database and subsequently to the application and the user.

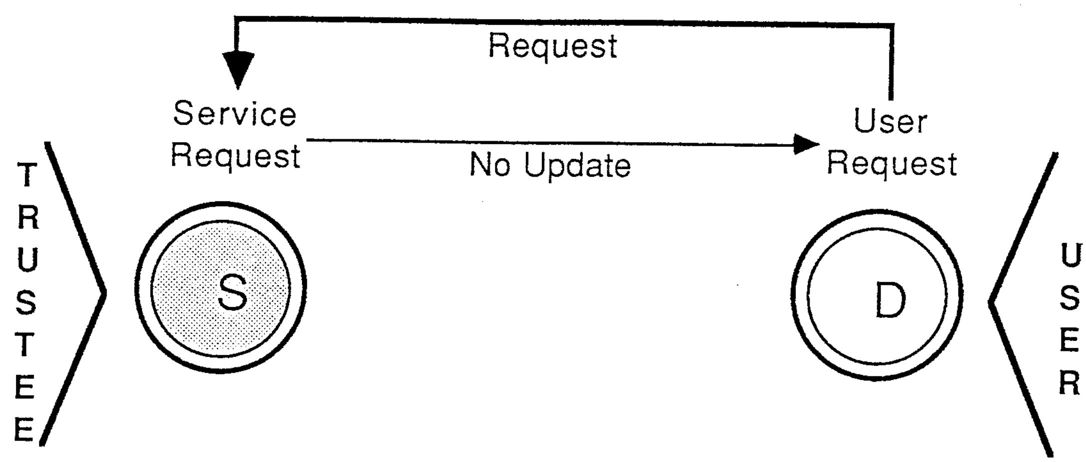


Figure 4.5 **Request for Update - No Update Available**

When a positive response to a request for propagation updates is given, a complex chain of events is triggered (Figure 4.6):

1) The user's application program requests data with a specified degree of currency from the destination database.

2) A check must be made with the destination update tracking database to see if the destination data set meets the requirements of the application.

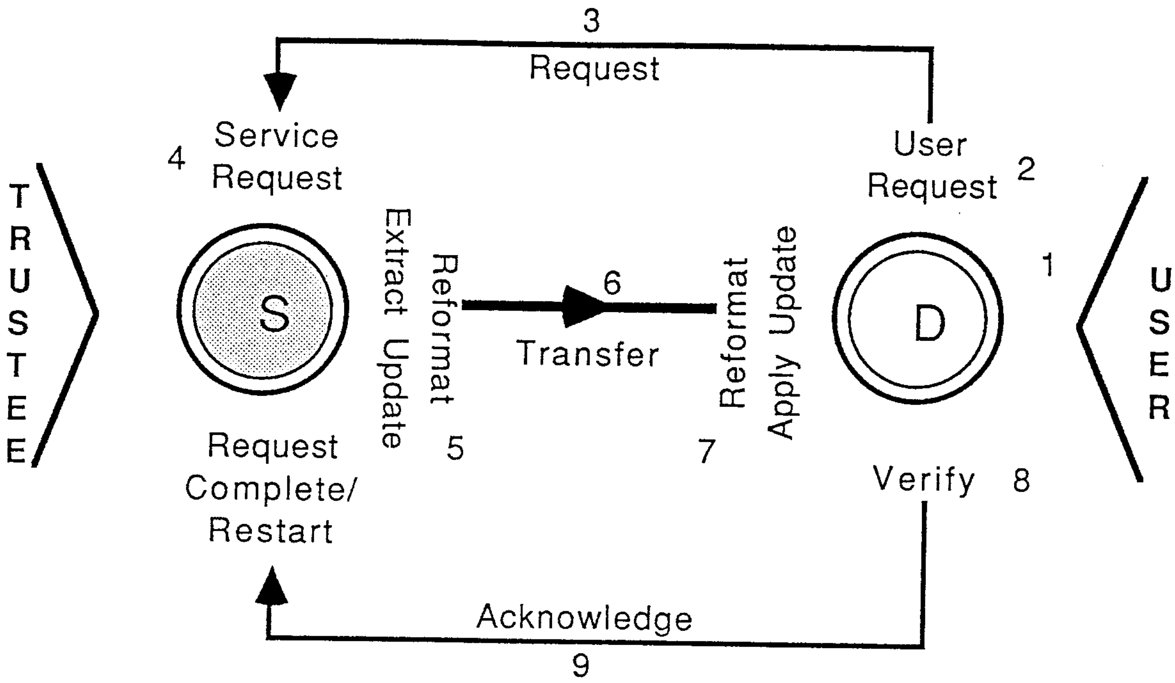


Figure 4.6 **Request for Update - Update Propagated**

3) The update process determines the source of the destination data set from the destination update tracking database. It then generates a request for the required update from the source land information system.

4) The request is transferred to the source update tracking database and the process determines whether a source update has occurred.

5) If, at this point, an update has occurred then the update process must extract the update from the source database.

6) Once the update is extracted, it may require reformatting and then be transferred (propagated) to the destination land information system.

7) The update process may be required to further re-format the update and must then apply the update to the destination database.

8) The update process then verifies that the propagation update was properly applied. The application program is blocked from using the updated database until the verification process is complete.

9) a) Upon successful verification, the update process sends an acknowledgement, to the source land information system, which will be recorded in the update tracking structure. As well, the application program will be informed via the destination update tracking database.
b) If verification is unsuccessful, then the update process will produce an error report and, upon approval of the destination land information system, re-submit the request for a propagation update from the source database. This action results in a repeat of the process beginning at step 4.

4.4 Approaches to Updating

Within the update process there are four approaches as to how source updates are recorded and how propagation updates are extracted, transferred, and applied. These approaches are depicted in Figure 4.7. Each approach involves trade-offs in the level of sophistication of the land information system required, volume of data transferred, and capacity for high update rates. The detailed criteria for the selection of an optimum data update policy are given in Chapter 7.

To this point in the development of the data update model, the nature of the data has not been of concern. The data being dealt with in a land information system has a graphics and/or an attribute component [ARCHER, 1986]. Depending upon the software system used for the land information system, these two may be highly integrated or quite separate. In order to accommodate the two types of data and the various levels of integration between them, both graphics and attribute data will be examined as they would affect the application of the four approaches.

The dual nature of the data in a land information system results in some difficult integrity constraint problems [GOH, 1987]. This is particularly true in systems where the integration between graphics and attribute data is low and in agencies where there are significant time lags which occur between the application of graphics and attribute updates. In applying the data update model these considerations must be dealt with.

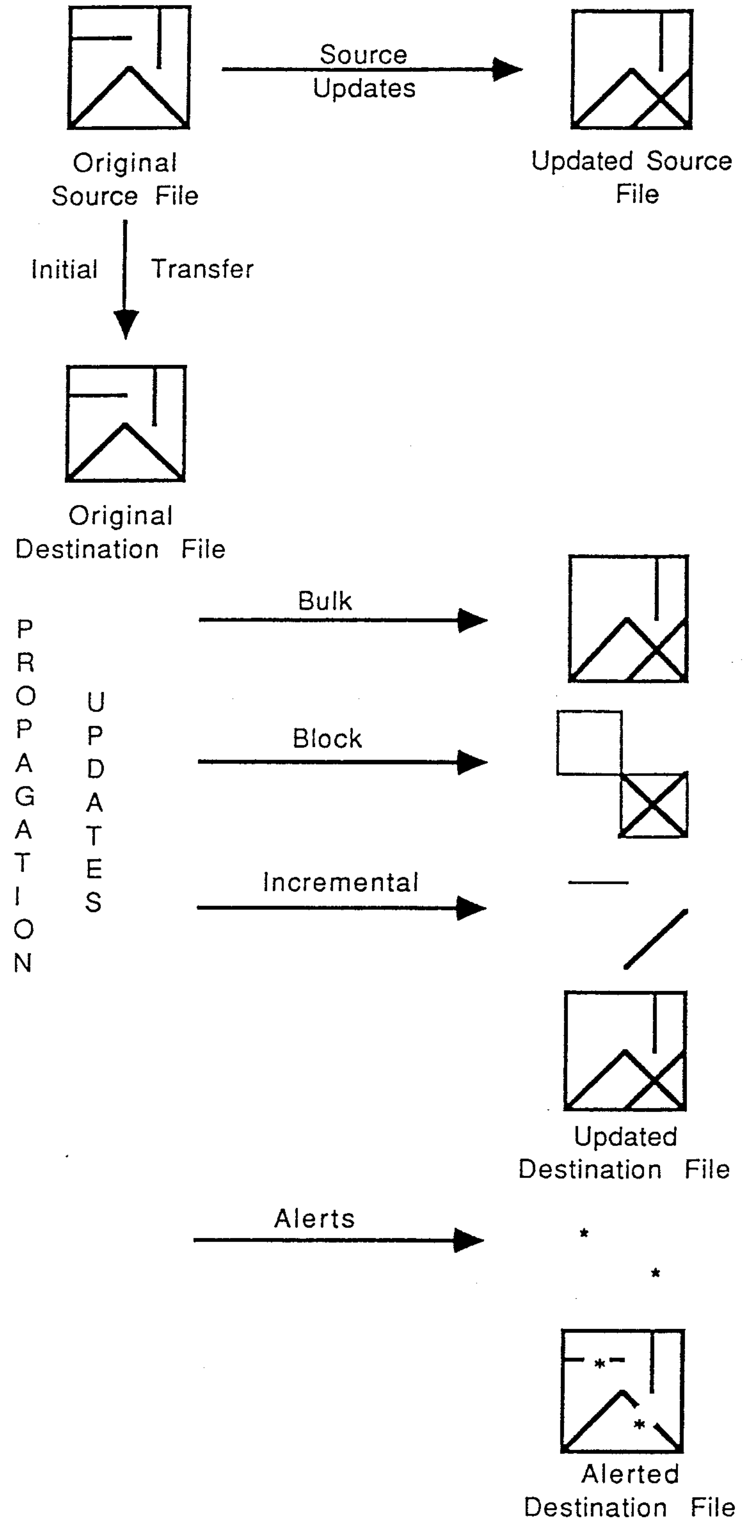


Figure 4.7 **4 Approaches to Updating**

4.4.1 Bulk File Update

The bulk file update approach accommodates source updates by recording the details and timing of the latest update to a file. There is no attempt to record the history of the update or to identify which feature has changed. When propagation updates are requested under this approach, and a source update has occurred, then the entire source file is transferred and replaces any existing data in the destination database for this particular theme and area. This approach assumes that the partitioning of the source and destination databases into files is identical or can be made equivalent. The approach is the same in transferring graphics and attribute files. Currently, this is the universally adopted approach [PEARSON et al., 1986].

The bulk file update approach requires a very low level of technical sophistication within the source and destination land information systems but it does result in high volumes of data being transferred. As well, the ease of operation is highly dependent upon the efficiency of reformatting data from the source system data structure to the destination system data structure.

It must be noted that three remaining approaches build upon an initial bulk file transfer. Therefore the bulk file update process must exist in any system of processing data updates.

4.4.2 Block Data Update

The block data update approach accommodates source updates by recording when the latest update to a block of data has occurred. It can be viewed as equivalent to a bulk file update but at a finer resolution. A block in a graphics data set refers to any user defined self-contained collection of features. This concept is evident in most land information systems, especially those which are based on cadastral parcels. In the South Australian Digital Cadastral Data Base (DCDB) there is a two tiered hierarchy of blocking referred to as super-blocks and macro-blocks [DEPARTMENT

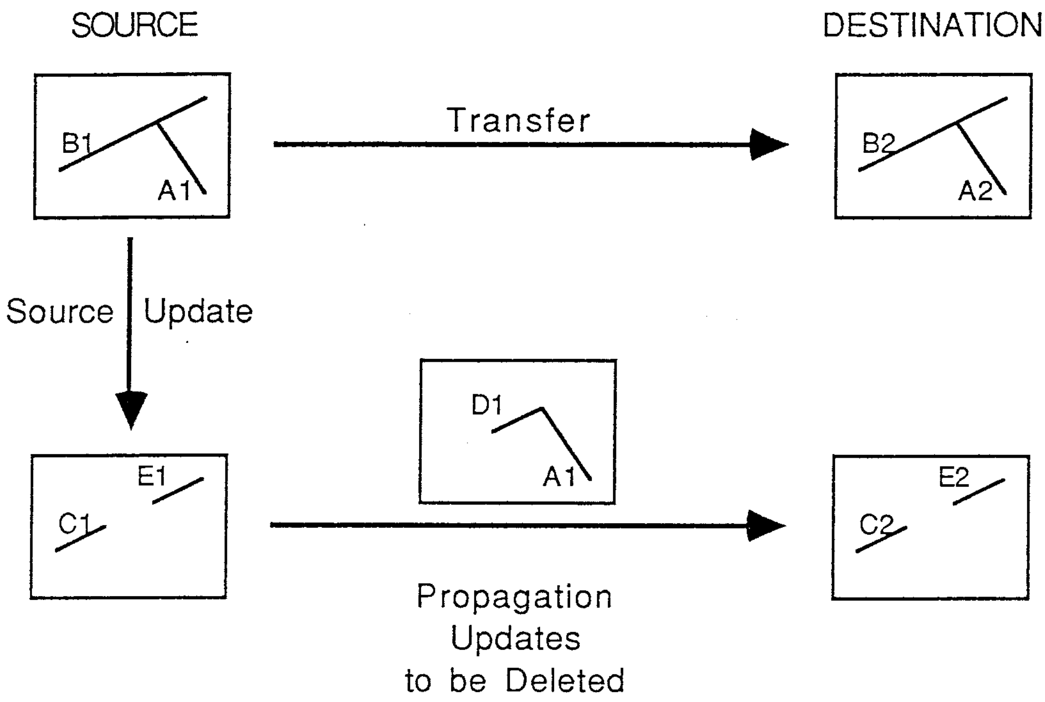
OF LANDS (SOUTH AUSTRALIA, 1986]. These blocks are based on data collection units and natural partitions in the data. Within an attribute data set, a block is equivalent to a record or tuple.

The propagation of block data updates occurs by transferring only those blocks of data from the source system which have changed. The update process must then delete the superseded blocks in the destination data set and replace them with the updated blocks from the source system.

This approach trades off increased system sophistication, over the bulk file update approach, for a decrease in the volume of data requiring transfer. This, of course, is dependent upon the characteristics of the updates. If the majority of updates occur in a limited number of blocks then the savings will be significant, but if an update occurs in each block then the bulk file approach would be more efficient. The nature of the updates and their impact on the optimum update approach is discussed in detail in Section 7.2.3.

4.4.3 Incremental Data Update

The incremental data update approach accommodates source updates by recording each individual change to the data set. A change is defined as an addition and/or a supersession of an entity in the data set. The term supersession is used to indicate the marking of an entity as being superseded or no longer current, as opposed to a deletion which indicates the elimination of the entity from the data set. The recording of changes to the source database at the entity level requires sophisticated source updating software in the source land information system. With graphics data, additions and supersessions are recorded for the primitive graphic elements: points and lines. With attribute data, changes are recorded at the sub-record, that is the data item, level. The recorded additions and supersessions are time-stamped to preserve the history of the file. Thus, the state of the data set at a particular time can be re-created and changes which have occurred since a particular time can be extracted and propagated to a destination system.



There is a 1:1 correspondence between A1 and A2 therefore deletion is straightforward. There is no 1:1 correspondence between D1 and B2 (D1 is a sub-set of B2) therefore deletion is complicated.

Figure 4.8 **Correspondence Between Source and Destination Graphics Data Sets**

The propagation updates consist of a collection of additions and supersessions which must be applied to the destination data set in chronological order. The application of the updates requires sophisticated software, keeping in mind that the systems which constitute the land information network are heterogeneous with respect to software, and most importantly, data structures. Graphics data is particularly difficult since an entity, in the source data set, which is to be superseded may occur as a sub-set of an entity or as a collection of entities in the destination data set (Figure 4.8). There is no guaranteed one-to-one correspondence between the entities in the source and destination graphics data sets. The entities in the destination data set which correspond to the supersessions must be deleted and the additions appended to the data set.

This approach trades off a high level of technical sophistication in the land information system's data structure and software for a dramatic reduction in the volume of data transferred in the previous approaches. The incremental update approach is well suited for situations requiring a high degree of timeliness, that is, the speedy propagation of updates.

4.4.4 Alert Update

The alert update approach is not a method of updating the destination data set but of placing markers in the destination data set which indicate that changes have occurred in the source data set in these areas. Within Queensland's Corporate working Map this is referred to as "noting" [HEBBLETHWAITE, 1986]. The source land information system must generate these update markers from source updates. The update markers are time-stamped and propagated to the destination system on a regular basis. The markers are simply appended to the destination data set. Markers of differing types can be used to denote characteristics of the source update being represented, e.g. addition, supersession, or the origin of the update. With graphics data sets, the markers are in the form of point symbols or polygons denoting the affected area. With attribute data sets, the markers are flags in the data records.

This approach offers minimal data transfer volumes, a low level of technical sophistication on the destination system, and the potential for very timely propagation of update markers. Depending on the characteristics of the data updates and the user application it may be an optimum approach to use in conjunction with, for example, the bulk file update approach.

4.4.5 Integration of Update Approaches

The implementation of the four update approaches does not follow clean theoretical categories. Within a particular update process a mix of these approaches will be required. According to the nature of the graphics and attribute systems, a block data update approach may be appropriate for one and an incremental data update approach may be appropriate for the other. More than one approach may be feasible within the same situation or the available technology may limit the options to a single approach. These factors and the criteria for selecting the most efficient approach will be dealt with in Chapter 7.

It is possible for one approach to source updating to generate more than one type of propagation update. The possible combinations are listed in Table 4.1 and shown in Figure 4.9.

Table 4.1 Conversion of Source Updates to Propagation Updates.

Source Update	Propagation Update
Bulk	Bulk
Block	Bulk
Block	Block
Block	Alert(?)
Incremental	Bulk
Incremental	Block
Incremental	Incremental
Incremental	Alert
Alert	Bulk
Alert	Block(?)
Alert	Alert

A bulk source update limits the system to generating a bulk propagation update while block source updates can generate bulk, block, and possibly alert propagation updates, as shown in Figure 4.9. If it is known that a particular block has been updated, then it is also known which file has been affected and a bulk update can be generated. It is also possible to generate an alert marker encompassing or at the centre of the updated block. This may be unsatisfactory depending upon the method of generating the alert and the nature of the blocking in the source file. This uncertainty is represented in Table 4.1 by a "?".

Incremental source updates enable the generation of all four types of propagation updates (Figure 4.9). By knowing which features have been added and superseded it is known which file has been updated, thence a bulk update may be generated. It can also be derive which blocks of features will be affected for block updates. Alerts may also be generated by examining the individual features in an incremental source update.

Alert source updates enable the generation of bulk, alert, and possibly block propagation updates. As in the previous cases, knowing that an alert appears in a file is sufficient to inform the operator that a bulk update is required. According to the location of the alert it may be possible, depending upon the method of generating the alert and the nature of the blocking in the source file, to determine which block has been updated in the source file. The uncertainty of this approach is represented in Table 4.1 by a "?".

From the above discussion it can be seen that some approaches to recording source updates enable the system to provide great flexibility in the generation of the corresponding propagation updates. Any application of the proposed model will certainly be a hybrid system, synthesizing its approach to adapt to the special circumstances it faces and users it serves.

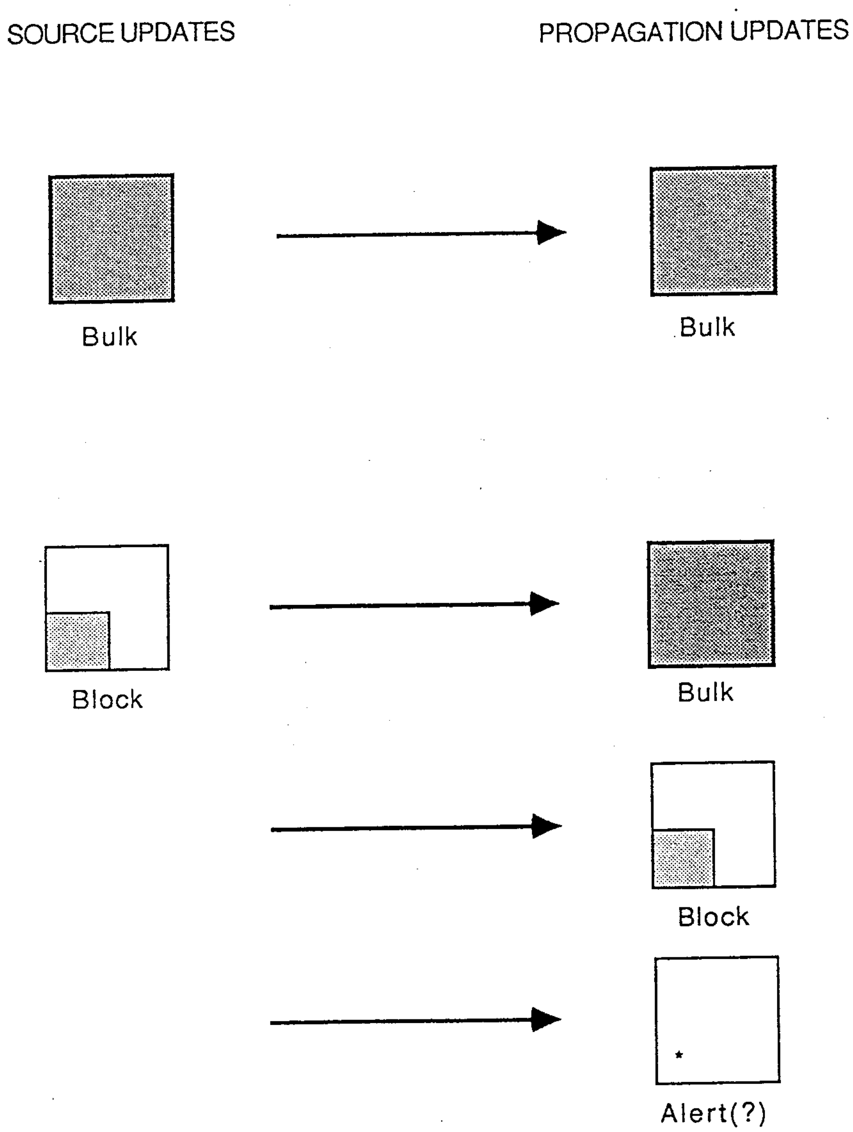


Figure 4.9 **Possible Propagation Updates Generated from Various Approaches to Source Updating**

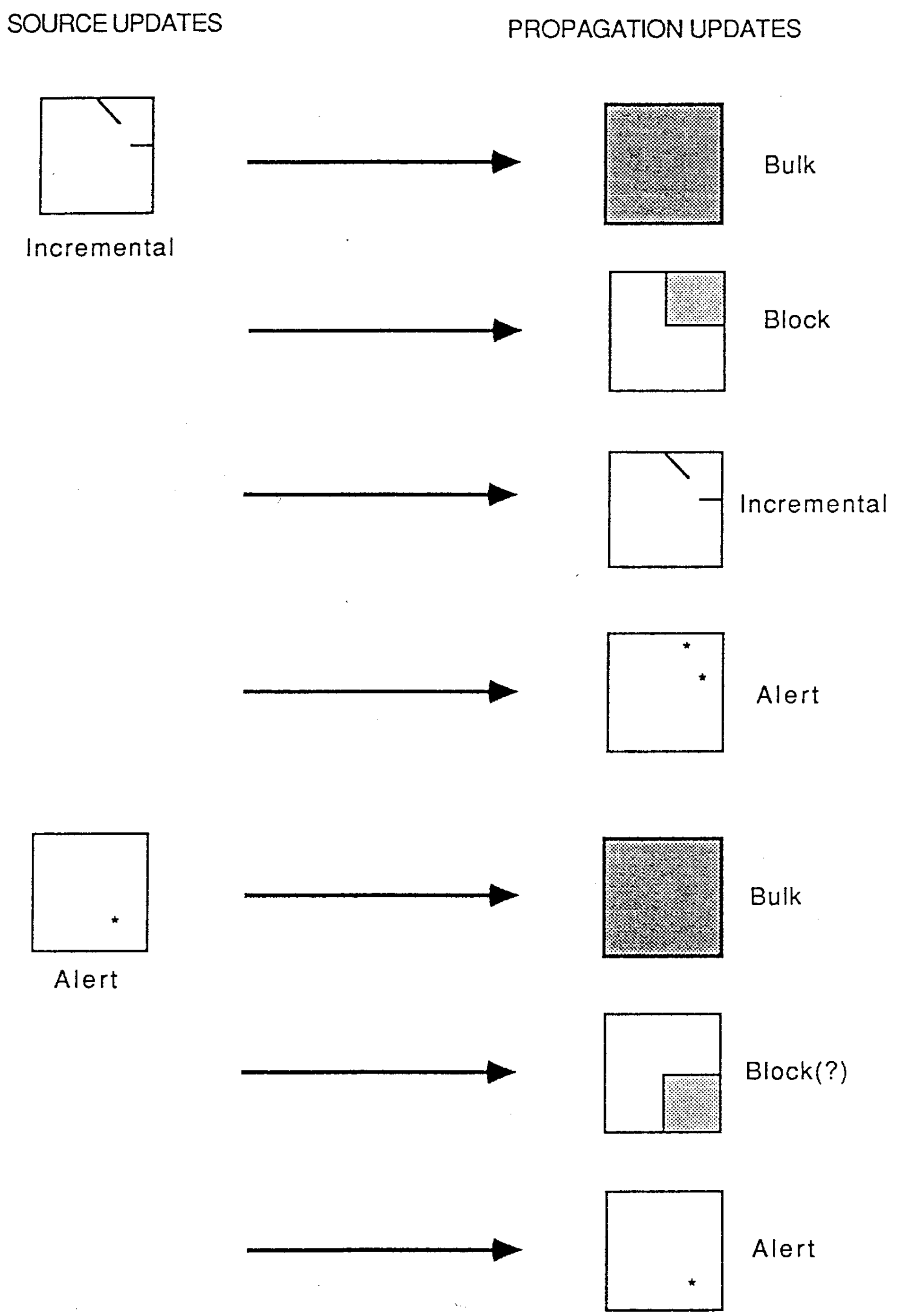


Figure 4.9 (cont.) **Possible Propagation Updates Generated from Various Approaches to Source Updating**

4.5 Summary

The model of data update in a land information network is based on a simplified view of a land information network as two inter-connected land information systems, one designated as the source and the other the destination. Each system must be capable of tracking the source of its data and the occurrence of any updates. The source system must be able to record any updates which are made to the source database.

The update process involves a destination system requesting updates from the source; the extraction, transfer, and application of these updates to the destination system; and finally the verification and acknowledgement of the updates.

There are four approaches to updating. Updating can be accomplished via 1) bulk file transfer, 2) transfer of blocks of data containing updated features, 3) transfer of only those features which have changed (incremental), and 4) transfer of markers or alerts which indicate a change has occurred in an area. Each approach has its own benefits and limitations and any application of this model will undoubtedly incorporate a combination of these approaches to best address the requirements of the particular land information network.

Chapter 5 FUNCTIONAL DESCRIPTION OF THE DATA UPDATE MODEL

The following sections present a functional description of the data update model. This description seeks to divide the model into functional units and to define the components and processes involved in each. Within a unit there are often a number of possible alternatives. An attempt has been made to describe the most viable approaches and not necessarily all possible alternatives.

These guidelines have been used to implement a prototype land information network update model with the ARC/INFO system. This specific implementation is discussed in Chapter 6. It is hoped that the functional description provides sufficient information to enable the implementation of the update model in other systems.

The functional units have been identified as:

1) Tracking;
2) Recording/extracting/applying;
3) Reformatting;
4) Transferring;
5) Verifying.

The recording, extracting and applying of updates is dealt with as one section because of the high degree of dependence the three functions have upon the update approach being used. As well, a number of tools common to each of the three functions are described.

5.1 Tracking Updates

The facility to track updates serves as the control structure for the update process. It is a database resident in each LIS, which records information about users of the land information system, files stored in the system, user requests for copies of, or updates to, data, and the status of these requests.

5.1.1 Files Required for the Tracking Database

Table 5.1 defines the files which makeup the update tracking facility. They are presented as relational tables. Each land information system in the network would maintain a tracking database which may be in computerized or analogue form.

Table 5.1 Relations in the Tracking Databases

Legend RELATION_NAME **Key_Field** Field Field	USER_REQUEST **Request_Number** User_Name Theme Area Update_Frequency Initial_Transfer_Date Last_Transfer_Date Update_Format Graphics_Transfer_Format Graphics_Transfer_Media Attribute_Transfer_Format Attribute_Transfer_Media Scale
USER **User_Name** Address Phone Email_Address User_Directory	
FILE_STATUS **File_Name** File_Type Source_Update_Format Creation_Date Last_Update_date Origin	REQUEST_STATUS **Request_Number** **Request_Date** **File_Name** Send_Date Acknowledge_Date Graphics_Status Attribute_Status

The USER file records information about network users. Each user has a unique identifier (User_Name) and information on contacting them. The User_Name is formed from two identifiers; the prefix identifies which system in the network the user resides on and the suffix is a unique identification for the user within that system. Contact may be by letter (Address), phone (Phone) or electronic mail (Email_Address). With systems connected via a digital network, electronic mail is ideal. Finally, the User_Directory stores the location of a directory on the system in which files bound for the user can be stored until transferred. There could also be information about the user's level of privilege to access various files or accounting information for charging/paying the user for data and updates requested/supplied.

The FILE_STATUS relation records information about files resident on the land information system. Each file has a name (File_Name) and type (File_Type). File types denote the structure of the file; for example, ARC/INFO, ASCII, or DLG. The Source_Update_Format specifies the way in which updates are recorded on the source file, i.e. bulk, block, incremental and alert. As discussed in Section 4.4.5, some source update approaches provide greater flexibility in the generation of propagation updates while others are more limited. The Source_Update_Format determines which propagation update formats are allowed. The Creation_Date and the Last_Update_Date record the time of updating. Each time the file is edited the Last_Update_Date will reflect the change. If the two dates are the same then no updates have been made to the file. Finally, the Origin denotes on which system in the network the file originates. If it is a source data set then its origin will match the prefix of the user's name. If it is a destination data set, the origin will yield the source system. As well, information regarding the level of privilege required to access the file may be stored.

The USER_REQUEST relation records information about requests for data and updates. The Request_Number is similar in composition to the User_Name. It identifies the request via two identifiers; the prefix denotes the system from which the request originates and the suffix is a number unique to the requesting system. The User_Name identifies the user requesting the data.

Theme and Area define the composition of the requested data with respect to content and spatial extent. They are used to determine which file or files the data or updates will come from. The Theme may be a file name; the Area may be a map sheet reference, co-ordinate window, or administrative area; or a facility may exist to translate between theme/area date and file names. This is dealt with more fully in Section 5.1.2.1.

The Update_Frequency defines the interval at which updates are required. The frequency may be periodic or on demand. Periodic update frequency is based on time or volume of source updates, e.g. daily, weekly, or after 20 update occurrences. By requesting an update after one source update occurrence the user is effectively being notified of any changes to the source data set. This type of request need only be lodged once and, at the designated interval, updates will be propagated to the user. An "on demand" approach requires the user to lodge a request whenever updates are required.

The Initial_Transfer_Date records when the initial bulk file transfer occurred. The Last_Transfer_Date records when the last update was transferred and applied to the file. When a periodic update frequency is operative the system measures the time of the period from the Last_Transfer_Date.

The Update_Format denotes the update approach being requested, i.e. bulk, block, incremental, or alert updates. It is conceivable that a user may desire differing update formats for graphics and attributes; e.g. bulk update of graphics data and block update of attributes. For simplicity's sake, in this research a uniform approach to both graphics and attributes has been assumed.

The Graphics_Transfer_Format gives the required data format for graphics data; e.g. ARC, DLG, AS2482, or hardcopy plot. The Graphics_Transfer_Media specifies the medium by which graphics data is to be transferred, e.g. direct digital communications, 1600 BPI tape, high density floppy diskette, or 24" by 36" mylar plotting film.

Similarly, the Attribute_Transfer_Format gives the required data format for attribute data, e.g. ASCII, dBASE, basic comma delimited (BCD), or hardcopy print out. The Attribute_Transfer_Media is identical to the Graphics_Transfer_Media although it is important to note that for any update request it need not be the same. A user may request graphics data on 1600 BPI tape and attribute data on a letter quality hardcopy print out. The Scale denotes the largest map scale at which the graphics data will be utilised, or in the case of a hardcopy plot the scale of the plot. It is acknowledged that in a digital environment, the use of map scale for the determination of data's fitness for use is fraught with problems [MOELLERING, 1985]. Nevertheless, map scale is currently widely accepted and understood, and will continue to be used until a more suitable approach is found.

The REQUEST_STATUS relation records the status of a request as it proceeds through the update process. The Request_Number is used to link this request to its corresponding USER_REQUEST record. The Request_Date denotes when the request was initiated. The File_Name specifies the file from which data or an update is requested. Together the Request_Number, Request_Date and File_Name form the key to the relation. A periodic update frequency will generate a number of requests with identical Request_Numbers and differing Request_Dates. Similarly, the mapping from Theme/Area specification to specific files may generate a number of requests with identical Request_Numbers and differing File_Names. Thus, a composite key of Request_Number, Request_Date and File_Name is required to provide a unique identification of a REQUEST_STATUS record.

The Send_Date stores the date on which the data or update is sent from the source to the destination system. Similarly, the Acknowledgement_Date records when the destination system acknowledges the receipt of the data from the source.

The Graphics_Status and Attribute_Status record the progress of the update request processing. The utilization of these two fields is discussed in Section 5.1.2.2. In general terms, if a request terminates abnormally then knowing the status of the processing will assist in pinpointing where the problem has occurred. In addition to the foregoing attributes, there may be a requirement to store information with regard to the size of the update file or the number of updates being transferred for accounting purposes.

5.1.2 Update Request Server

The update request server is a process, performed via software or manually, whereby requests for updates are recorded and serviced. The process utilizes the tracking database and is resident on each land information system in the network. It is the implementation of the update model described in Section 4.3.

There are two aspects to the update request server: 1) interface to the tracking database, and 2) processing of requests.

5.1.2.1 Interface to the Tracking Database

The interface to the tracking database handles the incorporation of the user requests and acknowledgements into the USER_REQUEST and REQUEST_STATUS files respectively. This can be accommodated via an operator keying the required data into the tracking database or via intelligent software processing a standard form completed by the user and transferred by electronic mail or other means. A standard form could be composed of keywords, values and delimiters which the software would interpret.

For example, an update acknowledgement could be as follows: "Request_Number=1234& Request_Date=01-OCT-1987& File_Name=KENSINGTON& Acknowledge_Date=01-NOV-1987& &" where "keyword = value" followed by a delimiter "&".

When a USER_REQUEST record is created, the interface must generate a corresponding REQUEST_STATUS record. Similarly, the interface must regularly scan through the USER_REQUEST file to determine if a periodic update frequency has been fulfilled. For example, if a week has elapsed since the last update request was generated for a USER_REQUEST record with a weekly frequency then a new REQUEST_STATUS record must be created.

In generating REQUEST_STATUS records from USER_REQUEST records a mapping from the Theme/Area information to a File_Name must be performed. In the simplest case the theme corresponds to one type of data, e.g. cadastral boundaries. It is conceivable that a user may require a theme which is a combination of types of data. For example, a theme called Highway_Design may contain existing roads, rivers, contours, cadastral bounds and soil types. The development of such a customized data product represents significant problems from the point of view of data updating. When an update is made to one file then the update must also be propagated to all of the themes in which the updated file participates. This problem of determining the extent of the impact of a change in one file upon other related files in a database has been studied by Goh [1987]. As of yet, no operational solution to the problem has emerged and this research considers only the simple case.

The Area field describes the spatial extent of the request data or update. This can be specified via:

1) Map sheet;
2) Co-ordinate window;
3) Co-ordinate of a centre point and radius
4) Street directory reference (e.g. Sydney's UBD or Gregory's);
5) Administrative area (e.g. local government area).

Again, the mapping of Theme/Area to File_Name can be performed manually or by intelligent software.

Finally, the interface must incorporate reporting facilities. These must be able to give the status of a particular update request, report on all requests at a particular stage in the update process, list any update requests which have failed, report on updates which have been transferred with no acknowledgement received, and identify update requests which are requiring an inordinate length of time to process. The advantage of using a database management system for the tracking database is that such reporting facilities are readily available, along with the capacity for additional ad hoc queries.

To summarize, the interface to the tracking database incorporates requests and acknowledgements into the tracking database, it generates REQUEST_STATUS records from newly entered USER_REQUEST records or on a periodic basis, and it provides reports on the status of the processing of requests.

5.1.2.2 Processing of Requests

The processing of requests is at the heart of the update request server. There are four basic steps in the process:

1) Determine whether updates have occurred on the requested file;
2) Extract any updates according to the specified update format;
3) Reformat updates to the required transfer format;
4) Transfer the updates to the destination system.

The pseudo-code for the processing of requests is given in Table 5.2.

Table 5.2 Pseudo-code for the Processing of Requests

```
Form a Universal Relation from the REQUEST_STATUS,
USER_REQUEST, USER, and FILE_STATUS.
Loop Processing each record in the relation until the end of file.
        Check that user has privilege to access file.
        If no privilege then, go to the next record.
        Check if an update has occurred since last
        update was transferred.
        If no update then, go to the the next record.
        Extract the updates according to the specified
        update format.
        If error during extraction then, go to next
        record.
        Reformat the updates to the specified transfer
        format.
        If error reformatting then, go to next record.
        Place updates on the specified transfer medium &
        transfer to the destination system.
        If error during transfer then, go to next record.
        Get next record.
End Loop.
Report results of processing
```

Processing is carried out by examining each REQUEST_STATUS record. It can be approached in two manners:

1) For each step, process all of the REQUEST_STATUS records before proceeding to the next step.

   or

2) For each record, perform all of the steps before proceeding to the next record.

The former approach ensures that users not requiring updates are notified quickly while the latter approach lends itself to the assigning of priorities to the processing of requests.

To begin, a universal relation composed of the REQUEST_STATUS, USER_REQUEST, USER and FILE_STATUS files is formed. Relational joins are performed between REQUEST_STATUS.Request_Number = USER_REQUEST.Request_Number, USER_REQUEST.User_Name = USER.User_Name and REQUEST_STATUS.File_Name = FILE_STATUS.File_Name. If a non-relational database is used a more procedural approach will produce the same result.

If user privilege and file protection information were included in the tracking database, a check could now be performed to determine whether the user has the right to access this file. If yes, then the process would proceed; if no, the process would complete for that particular request and notify the user.

Next, it must be determined whether a source update has occurred since the last propagation update was sent to this user. This is accomplished by comparing the FILE_STATUS.Last_Update_Date with the USER_REQUEST.Last_Transfer_Date. If the Last_Update_Date is earlier than the Last_Transfer_Date then no update is required and vice versa, if the Last-Update_Date is equal to or later than the Last_Transfer_Date then an update is required. A special case occurs when the Last_Transfer_Date is null, or equal to 01-JAN-0000. This indicates that no data transfer has previously occurred and that this is an initial request for data. Therefore, a bulk file transfer is required no matter what is specified in the USER_REQUEST.Update_Format.

If no update is required then a message to that effect is generated, transferred to the user on the destination system, the REQUEST_STATUS.Graphic_Status and REQUEST_STATUS.Attribute_Status are set to indicate no update is required, and the processing of this request is complete. If an update is required then the graphics and attribute status is set to indicate an update is required and processing continues. If an initial bulk transfer is required then that is indicated in the graphics and attribute status, and processing continues.

5.2 Recording/Extracting/Applying Updates

The recording, extracting and applying of updates are dealt with as one section because of the strong nexus between these functions and the update approach, i.e. bulk, block, incremental, or alerts, being considered. Recording is the process of noting when and where updates occurred in the source data. Extracting is the process of generating a copy of the source updates which, in turn, are applied to the destination data to bring it up-to-date.

5.2.1 Update Tools

The four basic approaches to recording, extracting, and applying updates can be implemented by a variety of methods. Each method varies as to the level of user intervention and system sophistication required. Each method will require differing tools or capabilities in order for it to be implemented.

The capabilities are as follows:

1) Graphically overlay files;
2) Edit features;
3) Append files:
4) Attach attributes to features;
5) Extract features based on associated attribute values;
6) Determine overlapping features between two files;
7) Polygon overlay.

Many of these tools are standard operations on commercial land information systems while others may be developed as application software. The implementor of the update request server will desire to make full use of the tools currently available to him and will only develop new tools if the methods which these new tools support yield sufficient advantages.

5.2.1.1 Graphically Overlay Files

Graphically overlaying files is the ability to view two or more graphics files simultaneously in registration. As well, it must be possible to distinguish between the files via colour or line type. This facility is useful in visually determining changes which have occurred to a file over time. As can be seen in Figure 5.1a, if a copy of the original file is superimposed on the update file, the features which have been added will be highlighted. Similarly, if the updated file is superimposed on the original, the features which have been deleted will be highlighted.

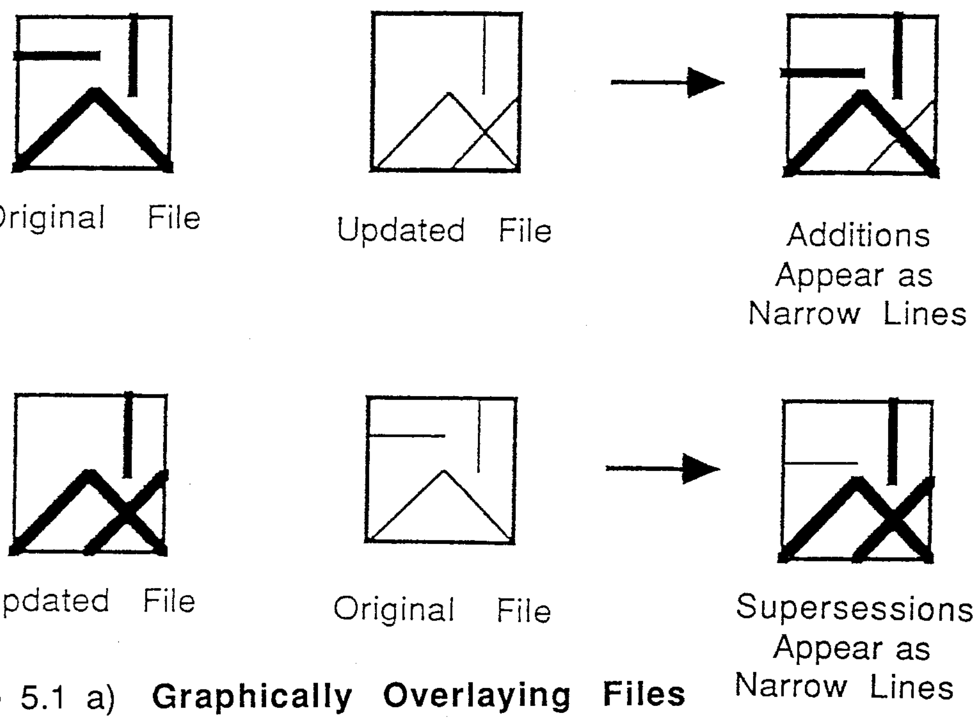


Figure 5.1 a) **Graphically Overlaying Files**

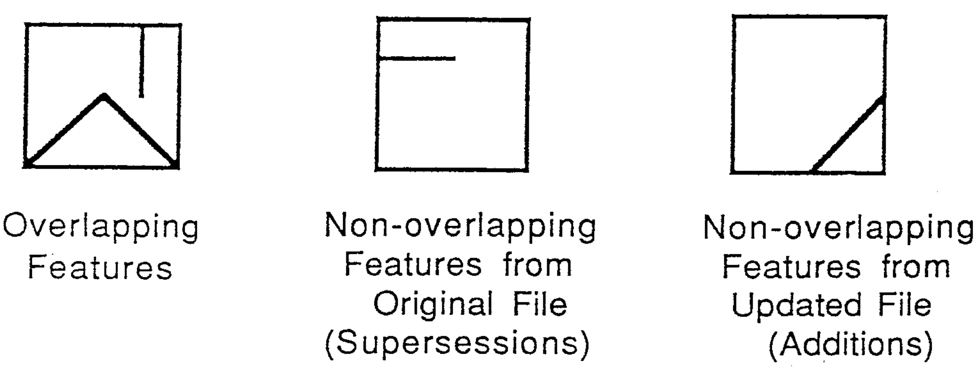


Figure 5.1 b) **Determining Overlap**

5.2.1.2 Edit Functions

Land information systems provide a variety of graphics editing capabilities. The following basic operations must be provided:

1) Add features:
2) Split lines;
3) Select features;
4) Copy selected features to a file;
5) Delete selected features.

The selection of features (3) should be accomplished by either interactive pointing with a cursor or by specifying a logical/arithmetic expression for attributes attached to the feature. It should be noted that in some systems the selection of features is incorporated into the copy and deletion operations and therefore there is no need for a separate operation to select features. Similarly, other systems may support these functions but as a multi-step operation. In this case it would be helpful to have a facility to build macros, or strings of commands which can be executed as a single operation.

These operations can be used to extract highlighted features or to create alerts. For example, two files could be graphically overlaid such that features deleted from the original file were highlighted. These features could be isolated, selected and copied to a deletions update file, or alerts denoting the changes could be added to an alerts file.

5.2.1.3 Appending Files

Appending files is the operation of combining two or more files into a single file. This operation is pivotal to being able to update a file with additional features or alerts.

5.2.1.4 Attach Attributes to Features

The ability to attach attributes to graphics features enables the explicit identification of individual features with one or more attributes. Specific land information systems may restrict the attributes to be of a fixed number and format or the system may allow the user to define as many attributes as he desires in the format of his choosing. The ability to freely define the attributes attached to features gives great flexibility in recording some types of incremental updates. In this case, the user can define attributes for the date the feature is entered in the database and the date on which the feature is superseded. If the system limits the user to defining a single numeric value for each feature, this can be used as the block identifier in the block update approach. It is assumed that if the system allows the user to define attributes associated with features, facilities will also exist to initialize these attributes, either during editing or as a batch process.

5.2.1.5 Extract Features Based on Associated Attribute Values

For systems which provide for the attachment of attributes to graphics features, the ability to extract features to a new file, on the basis of attribute values, is beneficial. For example, if the date on which a feature is entered into a data set is recorded along with the feature, it would be possible to extract all of the additions to the data set which have occurred since a particular point-in-time.

5.2.1.6 Determine Overlapping Features Between Two Files

Determining the overlapping features between two files, one designated as the overlapping file and the other as the base file, is done by selecting features in the overlap file one at a time, selecting all of the features in the base file which lie in the area covered by the overlap feature, and then comparing the overlap feature with the selected base features one at a time until a match is found or all of the selected features have been tested. The testing of the features for correspondence is not always a simple matter. As was presented in Figure 4.8, there is not necessarily a one-to-one correspondence between the same graphics feature in two different files. One feature may be a subset or a superset of the corresponding feature. As well, some manipulation and transformation operations may slightly alter the position of a feature, thus software to determine overlapping features may be required to match the features within a user specified tolerance.

An option which allows the overlap software to delete features, in one or both of the overlap or base files, which correspond, produces files which contain the non-overlapping features. These features represent the differences between the two files. As can be seen from Figure 5.1b, this can produce the addition and supersession features between an original and an updated file. Overlap determination also has application in verifying that an updated destination file is identical to the corresponding source file.

5.2.1.7 Polygon Overlay

The polygon overlay capability enables a file containing polygons to be superimposed upon another file such that each feature in this file is tagged with an attribute indicating which polygon it falls within. This finds application in the block update approach. A file containing polygons representing the block boundaries can be overlaid upon a file of features and each feature will be tagged with the block identifier, see Figure 5.2. This assumes that the land information system provides the facility to attach attributes to features.

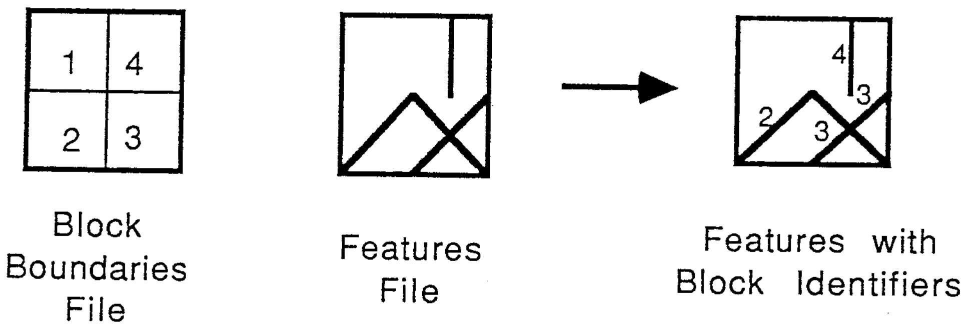


Figure 5.2 **Polygon Overlay Produces Features with Block Identifiers**

5.2.2 Implementing the Approaches

The following sections describe how the four approaches to updating can be implemented, utilizing the update tools previously defined.

A standard format has been utilized in describing each implementation. This is to facilitate the comparison of the approaches. The format is comprised of:

1) ***Title***
2) ***Purpose***
3) ***Requirements*** - outlines the general assumptions, tools and capabilities upon which the implementation is based.
4) ***File creation*** - defines the structures which must be defined and initialized during file creation to enable the recording and extraction of updates at a later point-in-time.
5) ***Recording*** source updates
6) ***Extracting*** propagation updates
7) ***Application*** of propagation updates
8) ***Limitations*** and special cases
9) ***Suitability*** of the implementation

The following implementations are examined:

1) Bulk Update
2) Block Update
3) Incremental Update with no facilities to attach attributes to graphics
4) Incremental Update with facilities to attach attributes to graphics
5) Alerts

It was necessary to divide the incremental update approach into two sections because of the significant difference in processing incremental updates which have attributes attached to the features and those which do not.

5.2.2.1 Bulk Updates

Title - Bulk Updates

Purpose - The purpose is to propagate updates by bulk transferring entire files. This is an initial requirement in any update approach and applies equally to graphics and attribute data.

Requirements - This approach requires no attributes directly associated with data items but utilizes the tracking database to record when a file has been updated.

File Creation - The file can be created in any fashion provided it is registered in the tracking database.

Recording - Updating can be carried out in any fashion provided that the fact that the file has been updated is recorded in the tracking database. There are two approaches:

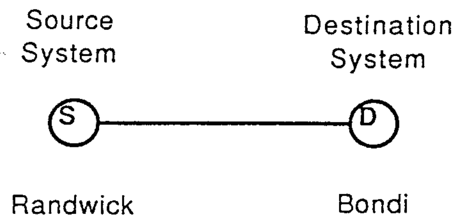
1) The edit software automatically updates the tracking database after edit.
2) The operator performing the edit manually updates the tracking database.

Extracting - An update extraction is not required, as the whole of the file is copied.

Application - Application of destination updates is achieved by deleting the existing superseded file on the destination system and replacing it with the propagation file.

Limitations - The limitation of this technique is the large volume of data which must be transferred. This will be aggravated if any necessary reformatting requires a significant amount of manual intervention.

Suitability - This approach is suitable in situations where there are a high volume of updates distributed across a file; or if facilities to attach attributes to specific graphics data items do not exist in the source system, thus precluding other approaches. Figure 5.3 gives an example of a bulk update.



Source Tracking Database (Randwick) FILE_STATUS

File_Name	**Last_Update_Date**	Origin
Traffic	**1-Jul-1987**	Randwick

Destination Tracking Database (Bondi) FILE_STATUS

File_Name	**Last_Update_Date**	Origin
Traffic	**1-Apr-1987**	Randwick

REQUEST FOR UPDATES

From: Bondi To: Randwick Date: 1-Aug-1987
Requests updates since: 1-Apr-1987
For file: Traffic
(details of update format, transfer media, etc.)

Since an update to the Traffic source file has occurred, namely 1-Jul-1987, a copy of the entire file will be sent to Bondi to replace the existing Traffic file. The Bondi tracking database will also be updated to become:

Destination Tracking Database (Bondi) FILE_STATUS

File_Name	**Last_Update_Date**	Origin
Traffic	**1-Aug-1987**	Randwick

Figure 5.3 **Example of a Bulk Update**

5.2.2.2 Block Update

Title - Block Update

Purpose - The purpose is to propagate updates for graphics files by transferring blocks of data which have been updated. This is an adaptation of the bulk data approach where a block is a subset of a file.

Requirements - This approach requires that an attribute be associated with each feature. This attribute records the identifier of the block to which the feature belongs. The selection of the definition of a block will depend upon the type of data and its application. Whenever possible, existing and/or meaningful groupings should be adopted; although artificial block boundaries will function effectively. Examples of blocking definitions are: local government areas, deposited plans, water sheds, or city blocks.

For the source updating and update extraction to occur, a file must be created which stores the block identifier of an updated block and the date of the update (Block Update List).

File Creation - Upon completion of the initial data input, the block identifiers must be defined for each feature. This can be accomplished in two ways:

1) During data input the operator initializes the block identifier attribute and it is recorded as the data is collected. The operator must remember to reset the block identifier each time he digitizes or edits a new block of data.

2) During data input the block boundaries are digitized first. These are written to a separate file (Block Boundary File) and topological polygons formed. Next, data is collected to fill in the block boundaries. When the digitizing and editing process is complete, the block boundary file is overlaid on the digitized data using polygon overlay techniques (Section 5.2.1.7). This will enable the block

identifier to be associated with each feature contained by that block boundary. This technique is less subject to error than the manual initialization of block identifiers during data collection.

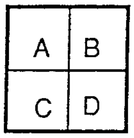
Recording - During source updating two functions must be performed: 1) new features added to the file must be tagged with the appropriate block identifier, and 2) record must be made of any block which is updated, through either addition or supersession, along with the time of the update.

The tagging of features is performed in the same manner as during the initial creation of the file. It can be by: 1) manually setting the block identifier, or 2) performing a polygon overlay with the block boundaries file.

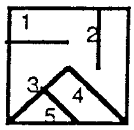
The recording of updated blocks' identifiers can be performed in the following ways:

1) Manually place the block identifier of the modified block into the block update list.
2) Provide a facility in the data editing software to record the block identifiers of features which are added and superseded.
3) Compare copies of the file before and after updating to determine the modified features, then perform a polygon overlay of the block boundaries and the modified features, and create a list of modified blocks based on the block identifiers associated with the modified features.
4) If a file is recording incremental source updates with attributes specifying addition and supersession dates, along with block identifiers, it is quite easy to create a block update list. Simply write out block identifier of features with addition or supersession dates since the last propagation update transfer date, to the block update list. Finally, eliminate any duplicate records from the block update list.

5.2.2.2 Block Update



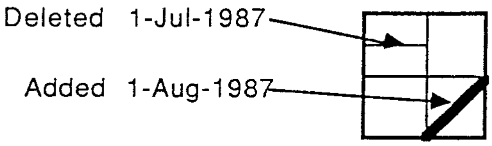
Block Boundaries File



Original File

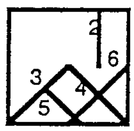
Original File
1-Jun-1987

Feature	Block
1	A
2	B
3	C
4	D
5	C



Block Update List

Date	Block
1-Jul-1987	A
1-Aug-1987	D

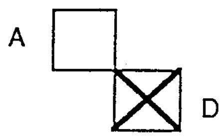


Updated File

Updated File
1-Sep-1987

Feature	Block
2	B
3	C
4	D
5	C
6	D

If a user requests all updates since 1-Jun-87 the update request server searches the Block Update List for affected blocks. In this case, blocks A and D would be extracted, transferred, and replace the existing blocks in the destination data set.



Propagation Updates

Figure 5.4 **Example of a Block Update**

Option 3 is the only option in which the operator performing the source updating requires no knowledge of the blocks being updated.

Extracting - The extraction of propagation updates involves the determination of the blocks which have been modified since the last update was applied to the destination file. This can be determined by examining the Block Update List which records the modified block identifier and the modification date. All block identifiers added after the desired date are extracted and duplicate entries removed. All of the blocks listed as modified are extracted from the source file (Section 5.2.1.5).

Application - The procedure to apply the block propagation updates is to determine the identifiers of the blocks in the propagation updates, delete these blocks from the destination file, and then to append the propagation updates to the destination file.

Limitations - The limiting factor in this approach is the attachment of block identifiers to features in the source file. If attachment is performed manually it is prone to error and time consuming. The alternative is to use polygon overlay which is a sophisticated, cpu intensive operation. It should be noted that there appear to be no special problems in updating data which falls along the boundary between two blocks provided a consistent set of rules for assigning the feature to a block are applied.

Suitability - This approach is suitable for systems which have limited capabilities in attaching attributes to graphics features but which require improved efficiencies over bulk updates. It is highly efficient for frequent and extensive updates which occur in localized areas. Figure 5.4 gives an example of a block update.

5.2.2.3 Incremental Update Without Attached Attributes

Title - Incremental Update with no facility to attach attributes to graphics

Purpose - The purpose is to propagate updates to graphics data sets by transferring only the changes (i.e. additions and supersessions).

Requirements - This approach is designed for systems without the facilities to attach additional attributes to graphics features. A copy of the file prior to update is required. The destination system's file must correspond to the copy of the source file prior to the source update. It is therefore assumed that all destination files will be updated on the same time cycle. If this is not the case then a copy of the destination file (which is in fact a copy of the source file prior to update) must be provided each time an update is required.

File Creation - The file can be created in any fashion.

Recording - Source updating can be performed in any fashion. It is assumed that the tracking database will record when a source file is updated.

Extracting - The updates are extracted by comparing the contents of the copy of the source file before updates were performed and the updated source file. Any features which appear solely in the updated file are additions and any which appear solely in the original file are supersessions.

All of the additions are copied to an additions file and, in the same manner, all supersessions are copied to a supersession file. This can be accomplished in two ways:

1) Manually, by superimposing each file, on a graphics screen and manually deleting overlapping features using a graphics editor (Section 5.2.1.1 and 5.2.1.2).
2) Automatically, by software which deletes overlapping features (Section 5.2.1.6).

Application - The application of destination updates is performed by concatenating the supersession file to the destination file and deleting overlapping features. This can be performed manually or automatically via software. This is followed by concatenating the addition file to the destination file. This completes the application of updates (Section 5.2.1.3, 5.2.1.1 and 5.2.1.6).

Limitations - The limiting factors in this approach are the update extraction and application steps. If performed manually it could be time consuming and labour intensive. If performed automatically, sophisticated software would need to be developed.

Suitability - This approach is suitable for systems with no facilities to attach attributes to graphics (Section 5.2.1.4) and which required a high update rate and low data volume.

5.2.2.4 Incremental Update With Attached Attributes

Title - Incremental Update with facilities to attach attributes to graphics

Purpose - The purpose is to propagate updates to graphics files by transferring only the changes (i.e. additions and supersessions).

Requirements - This approach requires the facility to attach attribute data to each graphical feature (Section 5.2.1.4). The attributes required would be the date on which the data was entered into the system and the date the data was superseded in the system. Optionally, since there is normally a time lag between data collection and entry into the database, the date of data collection and the date of data obsolescence would provide facilities for temporal analysis.

File Creation - Upon completion of the initial input into the source database, the addition date for each feature must be set to the current date and the supersession date set to infinity (e.g. 31-DEC-9999). This can be accomplished in two ways:

1) Initialize the attribute fields such that whenever a feature is added the dates are entered by default.

2) Once all data has been initially entered, perform a batch operation which will set the dates.

The choice of the most appropriate method is dependent upon the ease of implementation in the land information system concerned.

Recording - As source updating is performed, those features which are to be superseded will have their supersede date changed from infinity to the current date and those features to be added will have the current date for the addition date and infinity for the supersession date. There are two alternatives for the disposal of superseded features:

1) Superseded features remain in the updated file and can be used for time series analysis.
2) Superseded features are moved to a separate supersession file (some systems refer to this as the historical layer [DEPARTMENT OF LANDS (NEW SOUTH WALES), 1987]).

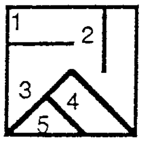
The latter may be required by systems which build topological relationships for all features in a file and of course superseded features in the updated file would interfere with this. This may also be desirable to unclutter the updated file and when temporal analysis is required the historical and current files can be merged.

Extracting - The extraction of propagation updates is simple and flexible. All addition features with the addition date later than the most recent propagation update transfer date (USER_REQUEST.Last_Transfer_Date) and a supersession date equal to infinity are required. All supersession features with a supersession date later than the most recent propagation update transfer date and an addition date earlier than the last propagation update transfer date are also required. If the feature code, addition and supersession attributes can be extracted with the update features, these can be used to validate the application of updates.

These extracts can be held in two forms:

1) Additions and supersessions in one file with attributes distinguishing the two.
2) A separate file for additions and a separate file for supersessions.

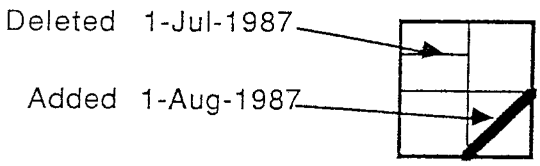
The choice of the form to hold the updates would depend upon how they will be required at the application of update stage.

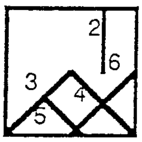


Original Source File

Original Source File
1-Jun-1987

Feature	Add_Date	Sup_Date
1	1-Feb-1987	31-Dec-9999
2	1-Feb-1987	31-Dec-9999
3	1-Feb-1987	31-Dec-9999
4	1-Feb-1987	31-Dec-9999
5	1-Feb-1987	31-Dec-9999



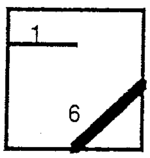


Updated Source File

Updated Source File
1-Sep-1987

Feature	Add_Date	Sup_Date
1	1-Feb-1987	1-Jul-1987
2	1-Feb-1987	31-Dec-9999
3	1-Feb-1987	31-Dec-9999
4	1-Feb-1987	31-Dec-9999
5	1-Feb-1987	31-Dec-9999
6	1-Aug-1987	31-Dec-9999

If a user requests all updates since 1-Jun-87 the update request server searches the source file and extracts all features with supersede dates and add dates since 1-Jun-1987. The updates are applied by deleting features corresponding to propagation updates with supersession dates less than 31-Dec-9999 and adding the remaining propagation updates.



Propagation Updates

Propagation Updates
1-Sep-1987

Feature	Add_Date	Sup_Date
1	1-Feb-1987	1-Jul-1987
6	1-Aug-1987	31-Dec-9999

Figure 5.5 **Example of an Incremental Update**

5.2.2.4 Incremental Update With Attached Attributes

Application - The application of the incremental propagation updates is performed by concatenating the additions to the destination file and then eliminating all features which correspond to the supersession file. The elimination process can be validated by checking to ensure that features to be superseded have feature codes and addition dates which match those of the corresponding entry in the supersession file. The elimination process can be performed either automatically via software or manually (Sections 5.2.1.3, 5.2.1.1 and 5.2.1.6).

Limitations - The limiting factor in this approach is the high degree of system sophistication required to attach attributes to features and the associated capabilities required to process this data. The update application if, performed manually, could be time consuming and labour intensive. If performed automatically, sophisticated software would need to be developed.

Suitability - This approach is suitable for systems with facilities to attach attributes to graphics features and which require a high update rate and low data volume. Figure 5.5 is an example of an incremental update with associated attributes.

5.2.2.5 Alerts

Title - Alerts

Purpose - The purpose of alerts is to propagate markers denoting where updates have occurred in the source file to the destination file.

Requirements - This approach does not require attributes to be attached to graphics features, although, this may be done to increase the information portrayed by the alerts. For example, attributes may be used to indicate the date of the update, type of feature updated, or whether the update is an addition or a supersession.

File Creation - The file can be created according to any of the procedures described for bulk, block or incremental updates. If no attributes are stored to indicate when individual updates occur, as with bulk updating, then a copy of the original file must be kept to ensure the detection of updates between the original and updated file can be performed.

Recording - In the same way, source updates can be recorded according to any of the three update procedures.

Extracting - Alert updates are generated rather than extracted. This is accomplished by detecting the changes which have occurred in the file. Changes may be detected by:

1) Comparing a copy of the original file with the updated file, either manually (Section 5.2.1.1) or via software (Section 5.2.1.6).
2) Selecting features whose attributes indicated that it has been added or superseded since the last update was transferred to the destination system.

Then, based on these features, graphical markers are recorded. This may be performed by manual editing or software. Markers may be represented as symbols placed at the centre of the affected feature, as polygons outlining the affected area, or both.

Application - Alerts are applied by simply graphically overlaying the destination data and the alerts in registration (Section 5.2.1.1) or by appending the destination file and the propagation alerts together into one file (Section 5.2.1.3).

Limitation - This approach is limited in that it is not a method to update data but merely a means of indicating that updates have occurred in the source data set. If the user requires the most up-to-date data and encounters an alert in his area of interest then he must request the updates in bulk, block, or incremental form.

Suitability - The simplicity of the alert approach enables it to be applied to any system. The low data volume of its propagation updates makes it suitable for source files with a high update rate. It is particularly helpful in situations where the only other approach to updating is the bulk update. Alerts can reduce the number of bulk update transfers by enabling the user to determine which areas have changed relative to his areas of interest within a file. Figure 5.6 is an example of an alert update.

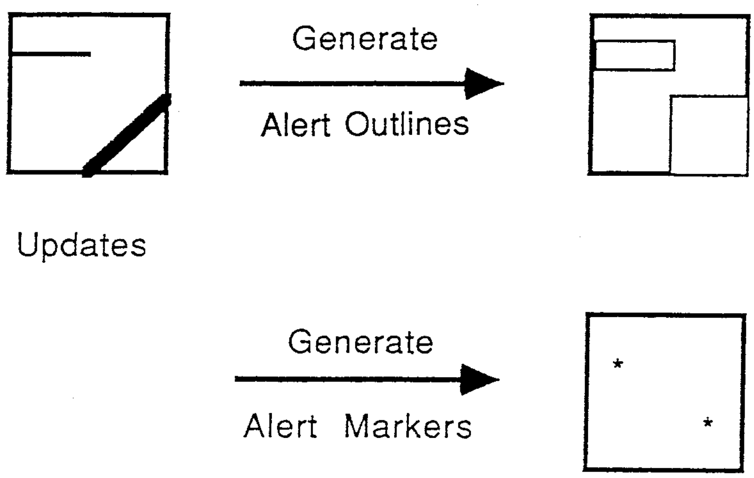


Figure 5.6 **Example of an Alert Update**

5.3 Reformatting Updates

The reformatting of updates refers to the translation of update data from the source system's data structure to an agreed upon data interchange format, and then from the interchange format to the destination system's data structure. Extensive research and discussion on the subject of standard data interchange formats has taken place in the land information systems community for the past ten years and although a number of "standards" have been produced no universal "standard" has emerged. The predominant formats used in Australia are the Australian Standard AS2482, Intergraph's Standard Interchange Format (SIF), MOSS, Autocad's DXF format, and simple ASCII files. Most agencies are looking to the work of the United States National Committee for Digital Cartographic Data Standards to establish a widely accepted standard [MOELLERING,1987].

Even if a standard data interchange format becomes widely accepted, this in itself is no guarantee that data can be exchanged without manual intervention. Differing techniques for the linking of attributes and graphics between a source and destination system or differing methodologies for the definition of common identifiers between a source agency and a destination agency may require the user to manually adapt the data received form the source for his purpose.

For example, the Sydney Metropolitan Water, Sewerage, and Drainage Board's Installed Facilities Information System (IFIS) has provided cadastral graphics and attribute data to the North Sydney Council's GENASYS system. Even though the GENASYS system has developed a customized piece of software to read the IFIS Interface Format, because of a difference in the method of uniquely identifying cadastral parcels, the GENASYS system requires the user to manually tag each parcel to associate the graphics data with the preexisting attribute database [WILSON,1987].

No matter if the tagging process is streamlined and efficient, it is still highly undesirable, being time consuming and error prone. If a bulk update approach were adopted then the tagging process would be required each time an update was propagated. The effort involved would be minimized by the block and incremental approaches. Thus, the reformatting of updates is a critical feature in a heterogeneous land information network and the extent to which it can be automated will have a significant impact on the update approach chosen.

In selecting a data interchange format for propagation updates a number of factors must be considered:

1) Support of required feature types;

2) Support of topological relationships;

3) Provision of meta-data;

4) Compactness;

5) Integration between attribute and graphics data.

The chosen data interchange format must support the types of features required by the destination system. For example, the Digital Line Graph (DLG) format does not explicitly accommodate point symbols, unless they are associated with a polygon [ALLDER et al.,1983]. Thus the DLG format would be inappropriate for data exchange requiring such point features.

Some data exchange formats provide facilities to record the topological relationships within the data, e.g. Digital Line Graph (DLG), most do not. Systems whose data structures utilize topology generally build the topological relationships from unstructured data rather than utilize the topological information stored in the data interchange format. Thus there would appear to be no advantage in the transfer of additional information which is generally not used. As is outlined in Section 5.5, even though topological information may not be used to reconstruct these relationships in the destination data structure it may play a significant role in the verification of updates.

With the advent of digital mapping, it has become possible to integrate graphics data from a variety of sources and with a variety of accuracies. Where once data represented by a map sheet was considered to be homogeneous with respect to accuracy and content we may now, without realizing it, view databases which are heterogeneous with respect to accuracy and content. This has led to the inclusion of data about data, or meta-data, in data interchange formats which describe the source and accuracy of the data being transferred. Definitive work in this area has been performed by the United States National Committee for Digital Cartographic Data Standards Working Group II on Digital Cartographic Data Quality [MOELLERING,1987]. By examining the meta-data the user can determine whether the data is fit for his application. This information would be highly desirable if the destination agency is not familiar with the source agency's data collection techniques or the data being exchanged is significantly heterogeneous with respect to accuracy and content.

The user will want to transfer data in as compact a form as possible. Due to the numeric nature of graphics data, binary files are more compact than ASCII files. As well, some approaches to padding partial blocks of data can lead to increased file sizes. This leads to a trade off between simplicity in creating/reading the file and the data volume. Where facilities exist to record data quality information and the data is homogeneous in nature, the source and destination agencies may agree to strip off the data quality meta-data to reduce file sizes.

Finally, the degree of integration between attribute and graphics data in a data interchange format may be an important consideration. Depending upon the application, a user may be interested in attribute only, graphics only, or both attribute and graphics data. Data interchange formats are spread over the whole spectrum of integration from:

1) Graphics with no attributes, e.g. AS2482
2) Graphics with limited attributes, e.g. Digital Line Graph (DLG)
3) Fully integrated graphics and attributes, e.g. United States' Federal Geographic Exchange Format, or IBM's Graphic Program Generator (GPG) Library Interface Format.

If required, it may be possible to store graphics and attribute data in differing formats and to provide a common identifier or key which enables the destination system to re-integrate the two data sets, when reading from the interchange formats to its own data structure. If this is not possible then, manual intervention would be required to re-establish the link between the attribute and graphics data.

5.4 Transferring Updates

The transfer of data updates involves the conveying of propagation updates, in some agreed upon interchange format, from the source to the destination system. This is performed by storing the updates on some medium and transporting the medium from one point to another. The medium may be, for example, hardcopy, floppy disk, magnetic tape, optional disk, or a communications network. The transport system may be, for example: personal delivery, courier, mail, telephone, satellite, or fiber optic cable. Developments in the area of data storage and computer communications are occurring at a staggering rate. As well, there are an endless number of combinations of medium and transport systems. Tanenbaum [1981a], expert in computer communications, has even suggested that St. Bernard dogs carrying floppy disks, rather than casks of brandy, may in some instances be an efficient data transfer method. Needless to say, the key requirement in the selection of a transfer method is that both the source and destination systems have the facilities to accommodate the method.

If a number of transfer options are supported between the source and destination systems then the criteria of timeliness and cost must be evaluated. Both of these will be affected by the volume of the data which is to be transferred. The volume of data, in turn, is dependent upon the characteristics of the updates and the approach to updating being used.

Figure 5.7 [Martin,1972] depicts the relationship between time, volume of data sent, and communications line speed. The data volumes required for data updating are derived from the headings of message delivery and bulk data transmission. As the requirements for timely updates increase or as data volumes increase, higher data transmission rates must be utilized to provide a satisfactory service to the users. The cost of higher transmission rates must also be borne by the users. Consultants to the New South Wales State Land Information Council [1987c] estimate that in excess of 50 Mbytes/day will be required to be transferred between the Bathurst and Sydney components of the state's Graphics DataBase (GDB) and that based on a 24 hour update window, this could initially be accommodated by a 48 Kbits/second communications line.

Currently communication networks between heterogeneous computer systems require vendor specific bridges and gateways into other systems. This leads to differences in the way in which databases are accessed and files transferred, depending upon the network being used. To simplify the data transfer process in this research, if an update is to be sent via a communication network a facility exists to store files in a user's directory on the source system and to then notify the destination system that they are available to be transferred. The user on the destination system is then responsible for the mechanics of the transfer. The notification may be sent via electronic mail which is further advanced in standard transfer techniques, e.g. ISO's X.400 messaging standard, than file transfer.

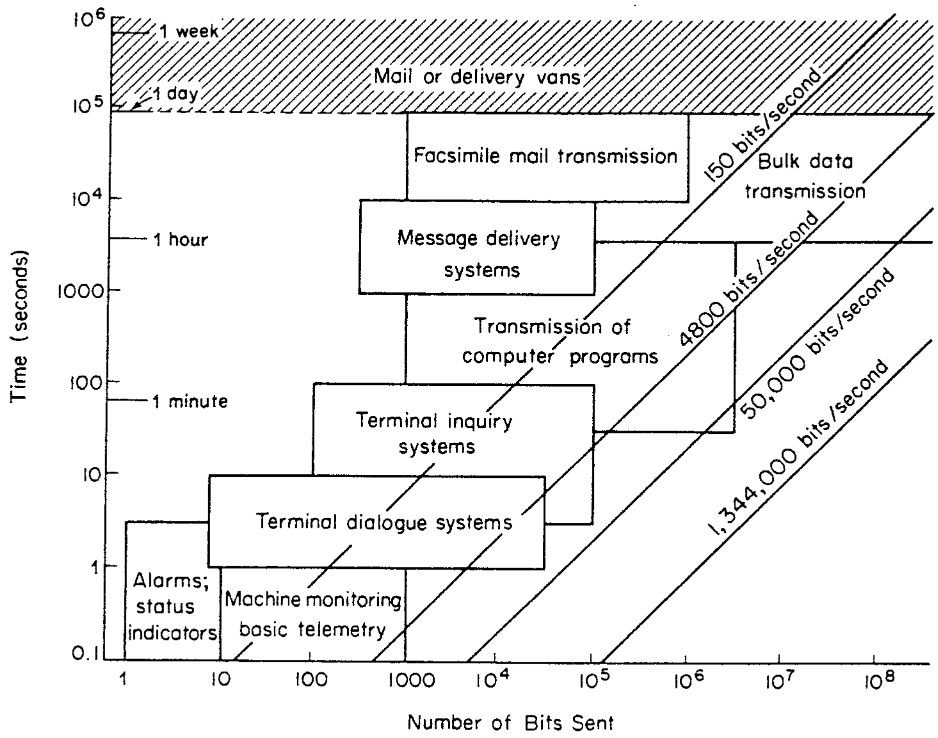


Figure 5.7 **Relationship Between Time, Volume of Data, and Communications Line Speed** [MARTIN, 1972]

As was outlined in Section 2.2.3, the Open Systems Interconnect standards' [TANENBAUM, 1981b] aim is to provide application to application communication, transparent to the user. This would enable, not only simplified messaging and file transfer, but update servers on source and destination systems to communicate easily and directly. Already the New South Wales State Land Information Council has made a firm commitment to the Open System Interconnect approach [STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES), 1987a].

In summary, there are a wide variety of storage media and transport options available for data update transfer in a land information network. These may be conventional, digital, or some combination of the two. As more data is required to be transferred more quickly the transfer cost will increase. The volume of data transferred may be reduced by re-examining the current data update approach in favour of a more efficient method. With the advent of the Open System Interconnect standards, we will soon see application to application communication between heterogeneous computer systems and this will make possible the ready implementation of fully automated data update processes.

5.5 Verifying Updates

The verification process seeks to detect any errors which may have occurred during the extraction of the updates from the source data set or the application of the updates to the destination data set. It is assumed that any errors in the transfer of the updates would be detected and rectified by the transfer process, although any which remain undetected should be evident during verification. As well, it is assumed that any errors detected during the update extraction or application would be resolved prior to verification. Thus verification is the fourth test in the update process, after the internal tests with extraction, transfer, and application.

The United States' National Committee for Digital Cartographic Data Standards (NCDCDS) Working Group II on Data Set Quality [MOELLERING, 1987] has published extensive work on the recording of data quality in data interchange formats. Their particular concern is that users receiving copies of data be capable of determining its fitness for the user's application. They propose five approaches to describing data quality:

1) Lineage;
2) Positional accuracy;
3) Attribute accuracy;
4) Logical consistency;
5) Completeness.

While the determination of whether data is fit for a particular application is outside the realm of the data update process, it is possible to utilize portions of the positional accuracy and logical consistency approaches to data quality to verify that data updates have been successfully performed.

Positional accuracy tests are used to determine if graphics features occur in the updated data set at the same spatial location as in the source data set. This can be checked via internal evidences, e.g. closure of traverses, or another approach is comparison to source. This follows the traditional approach of producing "check plots" which are overlaid on the source to determine discrepancies. This may be performed visually by graphically overlaying files (Section 5.2.1.1) or via software deleting overlapping features and yielding the differences (Section 5.2.1.6). The difficulty with this approach is that a bulk transfer of the source data set must be sent to the destination system for the check to occur. Due to the comprehensiveness of the check, it would be desirable for this to occur on a periodic basis, e.g. every three to six months, to determine whether there are flaws in the updating system.

Logical consistency tests "describe the fidelity of relationships encoded in the data structure of the digital cartographic data" [MOELLERING, 1987]. The tests are as follows:

1) Tests of valid values;
2) General tests for graphics data;
3) Specific topological tests.

Tests of valid values can be used to detect blunders resulting in unreasonable data values being entered. General tests for cartographic lines are given as [MOELLERING, 1987, Part III, pp.5]:

- "Do lines intersect only where intended?
- Are any lines entered twice
- Are all areas completely described
- Are there any overshoots or undershoots?
- Are any polygons too small, or any lines too close?"

Specific topological tests can be performed to certify that the source data set is "Topologically Clean". The updated destination data set can be subject to the same test. [MOELLERING, 1987,Part III,pp.5]

"For exhaustive areal coverage data transmitted as chains, or derived from chains, it is permissible to report logical consistency as "Topological Clean", under the condition that an automated procedure has verified the following conditions:

a) All chains intersect at nodes. (use of the exact case or tolerance must be reported)

b) Cycles of chains and nodes are consistent around polygons. Or alternatively, cycles of chains and polygons are consistent around nodes.

c) Inner rings embed consistently in enclosing polygon.

Considering the definition of polygon adopted in Part I (NCDCDS Definitions & References), conditions b and c require unique polygon identifiers."

If topological information has been transferred with the updates this may be useful in the topological checking process, especially in determining why particular errors have occurred. The topological test is rigorous but is applicable to only topologically structured data, i.e. polygons and networks.

To summarize, verification seeks to detect errors undiscovered in the extraction, transfer, and application steps. The two most suitable techniques are comparison to source and topological checking. The first technique requires large volumes of data to be transferred and the second is only valid for topologically structured data. The comparison to source approach should be carried out on a periodic basis to guard against flaws in the update process.

5.6 Summary

It is evident that the data update process has a wide variety of options for its implementation in a land information network. This requires users in the network to closely co-ordinate their activities. In the area of recording source updates, an inappropriate choice of approach in recording source updates can adversely affect a large number of users requiring propagation updates from the source system. In communicating between tracking databases, there is a need to standardize message formats. In reformatting and transferring, there must be agreed upon standard data interchange formats, storage media, and transfer techniques.

Even though the objective is to create an update process which is as "transparent to the user" as possible, it must be realized that a sophisticated system will require more intense co-ordination and co-operation than is evident in existing systems. This will require a commitment to the welfare of the land information network, as a whole, which transcends a particular users's interests.

Chapter 6 PROTOTYPING AND TESTING WITH ARC/INFO

The following sections describe the prototyping and testing of the data update model. The prototype was developed to determine the feasibility of implementing the data update model from the functional description and to determine whether the postulated advantages of each approach to data updating would be realized. Since the prototype is based on the functional description, it will be described according to the same categories as given in Chapter 5.

The model was implemented using the ARC/INFO geographic information system software, on a Digital Equipment Corporation VAX 11/785 under VMS. ARC/INFO is a well established product, developed and marketed by the Environmental Systems Research Institute of Redlands, California [ESRI, 1985]. It provides data input, analysis, and output facilities, and utilizes a vector data structure supporting full polygon and network topologies. As well, the INFO relational database management system, from HENCO, is integrated with the graphics system to support attribute manipulation [HENCO, 1985]. ARC/INFO provides a wide selection of "tools" for the user to develop customized applications. It is this last feature which makes the system particularly suitable for the prototyping of the data update model defined in this research.

The data update model was prototyped by implementing both the source and destination systems using the ARC/INFO software; thus the assumption that systems in the land information network will be of a heterogeneous nature, with regards to hardware and software, has been violated. This has been done knowingly in order to simplify and speed up the prototyping process. It is conceded that there will undoubtedly be situations occurring in a truly heterogeneous environment which have not been encountered in this research, but this is the case with any system prototype.

Each of the following sections contains a description of how the particular functional unit was implemented with the ARC/INFO system and the results of any testing performed. An attempt has been made to minimize the repetition from Chapter 5 and to minimize the use of ARC/INFO terminology to enable readers who are unfamiliar with the ARC/INFO system to benefit from the discussion.

Test results for times will normally be quoted in cpu seconds. Elapsed times for tests were recorded but due to the heavy and erratic load on the host computer, resulting from a large number of time sharing users, the times varied widely and did not permit a fair comparison of values. Most operations were of a batch nature but some interactive operations would have benefited from a consistent environment in which elapsed times could be reliably compared.

Test results for data volumes will normally be quoted in 512 byte blocks in ARC/INFO interchange format. The dual nature of the ARC/INFO software makes it difficult to readily determine data volumes of files in their normal structure. A coverage, as ARC/INFO files are referred to, is comprised of files in the graphics system, ARC, and the database management system, INFO. The ARC/INFO interchange format combines these files into one fixed length record, ASCII file. Thus coverages in ARC/INFO interchange format will not be the same size as the original coverages from which they are derived, but they will permit ready relative file size comparisons.

It will be left to Chapter 7 to integrate the test results into a cohesive criteria for the selection of an optimum update policy for a particular land information network.

6.1 Tracking Updates

The facility to track updates has been implemented with VMS command language procedures and INFO programs. It serves as the control structure for the update process.

6.1.1 Files Required for the Tracking Database

Tables 6.1 to 6.4 list the relations in the ARC/INFO tracking database. They closely follow those listed in Table 5.1.

Table 6.1 **FILE_STATUS Relation as Defined in the ARC/INFO Tracking Database**

DATAFILE NAME: FILSTS
8 ITEMS: STARTING IN POSITION 1

COL	ITEM NAME	WDTH	OPUT	TYP	N.DEC
1	FILNAM	16	16	C	-
17	FILDIR	24	24	C	-
41	FILTYP	8	8	C	-
49	CREDAT	8	10	D	-
57	UPDDAT	8	10	D	-
65	PROT	4	4	C	-
69	LIFE	4	4	C	-
73	RATE	4	4	C	-

Table 6.2 **UPDATE_STATUS Relation as Defined in the ARC/INFO Tracking Database**

DATAFILE NAME: UPDSTS
8 ITEMS: STARTING IN POSITION 1

COL	ITEM NAME	WDTH	OPUT	TYP	N.DEC
1	REQNUM	8	8	I	-
9	REQDAT	8	10	D	-
17	SNDDAT	8	10	D	-
25	ACKDAT	8	10	D	-
33	FILNAM	16	16	C	-
49	SIZE	8	8	I	-
57	GSTAT	8	8	C	-
65	ASTAT	8	8	C	-

Table 6.3 **USER Relation as Defined in the ARC/INFO Tracking Database**

DATAFILE NAME: USER
7 ITEMS: STARTING IN POSITION 1

COL	ITEM NAME	WDTH	OPUT	TYP	N.DEC
1	USRNAM	8	8	C	-
9	ADDRESS	16	16	C	-
25	PHONE	16	16	C	-
41	PRIV	4	4	C	-
45	ACNT	8	8	C	-
53	USRDIR	24	24	C	-
77	USREMA	8	8	C	-

Table 6.4 **USER_REQUEST Relation as Defined in the ARC/INFO Tracking Database**

DATAFILE NAME: USRREQ
13 ITEMS: STARTING IN POSITION 1

COL	ITEM NAME	WDTH	OPUT	TYP	N.DEC
1	REQNUM	8	8	I	-
9	USRNAM	8	8	C	-
17	THEME	8	8	C	-
25	AREA	8	8	C	-
33	FREQ	4	4	C	-
37	INITRN	8	10	D	-
45	LSTTRN	8	10	D	-
53	UPDFMT	4	4	C	-
57	GTRFMT	4	4	C	-
61	GTRMED	4	4	C	-
65	ATRFMT	4	4	C	-
69	ATRMED	4	4	C	-
73	SCALE	8	8	I	-

The relations in Tables 6.1 to 6.4 are described in INFO's data definition format where ITEM NAME is the field name, WDTH refers to the field size in characters and TYP is the field type (C-character, D-date, I-integer). The variations between Tables 6.1 to 6.4 and Table 5.1 are due to improvements made in the functional description, which were discovered after testing was well advanced, or due to the inclusion of functions which are peripheral to the data update process. The description of the relations represents the files as they appeared in the tests. The FILSTS.Fildir has been changed to FILE_STATUS.Origin and a FILE_STATUS.Source_Update_Format has been added to the functional description.

All of the source files in ARC/INFO were in an incremental source update format in order to give the greatest flexibility in generating propagation updates. The FILSTS.Prot, FILSTS.Life, FILSTS.Rate, UPDSTS.Size, USER.Priv, and USER.Acnt fields were included for privilege and accounting purposes and were only partially implemented.

6.1.2 Update Request Server

6.1.2.1 Interface to the Tracking Database

The interface to the tracking database has been established by using standard INFO operations. The operator keys data directly into the USER_REQUEST relation using INFO and manually generates the REQUEST_STATUS records. The development of a small application program to create USER_REQUEST records from a standard electronic mail message would not be difficult.

It was assumed that the USER_REQUEST.Theme corresponded to a file and that the USER_REQUEST.Area covered the extent of the file. The Theme/Area to file mapping facility could be readily implemented using the ARC/INFO Librarian software. The Librarian is a cataloguing system for a map library which allows the user to extract portions of the library, defined by a layer and window, as if the library were one continuous map sheet.

The reporting of the status of particular update requests or summary reports of processing are generated with standard INFO report features.

6.1.2.2 Processing of Requests

The processing of requests is performed by a command language procedure (PROUPD) which runs a series of INFO programs. These programs, in turn, create a temporary command language procedure (TEMP) which executes the update operations, e.g. extracting, reformatting, transferring. The PROUPD procedure can be scheduled in the VAX batch queue such that it is run on a periodic basis, e.g. every 4 hours. A weakness in using the TEMP procedure is that any errors which occur during the execution of TEMP are not reported to the INFO program. For the purpose of prototyping, this is acceptable but in a full function implementation, the process generating the update operations must be able to test the success of each operation as it executes.

Tables 6.5 to 6.9 contain the pseudo-code for the processing of requests. Communication to the user is through the VAX mail facility. The operator performing plots, printouts, and creation of magnetic media copies of data updates is informed by running reports from the tracking database.

Table 6.5 **Pseudo-code for the Processing of Requests**
PROUPD - PROGRAM TO SERVICE USERS' UPDATE REQUESTS. Relate the files required to compare the user's privilege and the file's protection. Set up a file to hold mail commands. Check the user's privilege vs file protection. PRIV.CHK Check to see if an update to the file has occurred since the user got his last version. UPD.CHK Filter the qualifying update requests to extract update data, reformat the filtered requests, and place in the user's directory. FILTER.EXT Transfer the extracted updates. TRANS.UPD End

Table 6.6 **Pseudo-code for the Checking of a User's Privilege**
PRIV.CHK Subroutine to CHecK the user's PRIVilege vs the file protection. Loop examining each user request. Compare privilege and file protection, If no privilege, send an appropriate message to the user. End loop. Display summary of the processing. End

Table 6.7 **Pseudo-code for Checking Whether an Update has Occurred**
UPD.CHK Subroutine to CHecK whether an UPDate has occurred since the user received his last version. Select all update status' which have the required privilege. Loop thru update status' checking to see if an update has occurred since the last transfer of data. *Note: The first request for update data is a bulk file transfer no matter what update format is requested.* If no update required, send an appropriate message to the user. End loop. Display summary of the processing. End

6.1.2.2 Processing of Requests

Table 6.8 **Pseudo-code for Filtering and Reformatting Updates**

FILTER.EXT
Subroutine to FILTER updates from a file to produce reformatted EXTracts.

Select the requests to be filtered.

Construct the source to temporary & destination directory/file names

If graphics or attributes are required then.

  Generate commands to filter out the required data, based on the update format.

  If bulk update, simply copy the file from source to temporary.

  End if - Bulk update.

  If block update, extract blocks since last update was transfer

  End if - Block update.

  If Incremental update, extract incremental changes since last update was transferred.

  End if - Incremental update.

  If Alert update, generate alerts to mark changes since last update was transferred.

  End if - Alert update.

  If reach this point then unsupported update format type.

  End if - unsupported update format type.

Commands to reformat the filtered data updates.

If update data exists.

  If update data exists in ARC/INFO format then.

    If graphics output format is ARC/INFO then.

      Will output both graphics and attributes from the input file

Move from temporary file to user's file.

End if - output file type ARC.

If graphics output format is DLG then.

If attribute output format is DLG then.

Output graphics and then append the attributes.

Else if another attribute output format is required then.

Insert the USER-ID in the DLG file as a link to the attribute file.

Else, output DLG graphics only.

End if.

End if - output file type DLG.

If no graphics output is required then.

End if - no graphics output required.

If reach this point then specified graphics output format is unsupported.

End if - graphics output format unsupported.

If attribute format is supported by INFO then.

INFO supports BASIC comma delimited (SDF), ASCII (ASCI), dBase/Lotus (DIF), and others not used here.
NOTE: A user can define his own format in INFO.

End if - attribute format.

If no attributes are required then.

End if - no attributes required.

If reach this point then specified attribute output format is unsupported.

End if - attributes output format unsupported.

Clean up the temporary file before finishing

Else, update graphics format specified is unsupported.

End if - source file type ARC/INFO.

End if - update data exists.

End if - graphics or attribute updates required.

Display summary of the processing.

End

Table 6.9 **Pseudo-code for Transferring Updates**

TRANS.UPD
Subroutine to set-up the update transfer requests.

Select all of the user requests with extracted updates to be transferred.

If the data is to be transferred as hardcopy then.

Via mail, instruct the operator on plotting and/or printing and shipping the data.

Else, if the data is to be transferred on magnetic media then.

Via mail, instruct the operator on the desired magnetic media and shipping of the data.

Else, if the data is to be tele-communicated then.

Via mail, instruct the user on how to access and download a copy of the data.

Else, error condition.

End if.

Display summary of the processing.

End

6.1.2.2 Processing of Requests

The processing of requests system functions satisfactorily. The next step in enhancing its operation would be the implementation of facilities to process messages from other tracking databases automatically, without the operator entering records in the tracking database. This could be accommodated by either programming the update request server to periodically read messages from an electronic mail box or by implementing process to process communication between update request servers on different systems in the land information network.

6.2 Recording/Extracting/Applying Updates

The facilities to record, extract, and apply updates have been implemented with VMS command language procedures, ARC graphics tools, INFO database programs, and custom written FORTRAN graphics and database tools.

6.2.1 Update Tools

The tools to support the update process are a mixture of ARC/INFO standard operations and customized functions. It is desirable that a land information system support all of the update tools, but if it does not, the variety of approaches to data updating allows the selection of an approach which can be accommodated by any land information system, without the development of customized tools. In this research, all of the tools were required for the purpose of testing the four update approaches.

6.2.1.1 Graphically Overlay Files

The ability to graphically overlay files is a fundamental ARC/INFO operation (ARCEDIT's Editcoverage & Backcoverage commands, and the ARCPLOT commands). It is possible to distinguish between the overlays by colour or line type and, as well, the user can select the features to be displayed within a layer (ARCEDIT's Drawenvironment & Backenvironment and the ARCPLOT commands).

6.2.1.2 Edit Functions

All of the editing functions required by the update process are provided by ARCEDIT, the program for interactively editing ARC/INFO coverages. Features can be added (Editfeature & Add commands), lines split (Editfeature Arc, Select & Split commands), features selected (Select command), selected features copied to a file (Put command), and selected features deleted (Delete command).

It should be noted that these functions normally require multiple ARCEDIT commands to carry them out. ARCEDIT does provide a facility to execute commands which are stored in a file (Command command) and thus supports a macro facility. This enables these functions, made up of multiple ARCEDIT commands, to be executed as a single operation.

The ARCEDIT command to select features is particularly powerful. It allows the selection of one or many features with a cursor defining the feature's location by a point or box. As well, the full functionality of the INFO database management system can be used to select features based on attributes associated with the graphics features.

6.2.1.3 Appending Files

ARC/INFO provides a facility to append, or merge, coverages together (Append command). This command appends files containing like features together, e.g. arcs, and preserves their associated attributes; or it will append files containing any feature type but drop the associated attributes. This is unacceptable for the data update process. We will want to append updates containing differing feature types, e.g. points and lines, to files containing similar features and retain the associated attributes. To resolve this limitation a custom FORTRAN program was written which utilizes INFO and ARC subroutines to enable two files containing differing feature types and associated attributes to be appended together without losing the associated attributes. This program is executed via a command language procedure entitled Adconc (Additions and Deletions CONCatenation procedure).

6.2.1.4 Attach Attributes to Features

ARC/INFO provides extensive facilities for the definition and attachment of attributes to graphics features. Attributes can be added singularly (Additem command) or a table of attributes created and joined to the graphics features via a common identifier (Joinitem command). The major INFO field types are supported and the size limit is 4096 bytes per record. Thus ARC/INFO easily supports addition/supersession dates and super block identifiers associated with each feature, as required in the incremental and block update approaches.

Facilities exist to initialize attribute values during data input (ARCEDIT New command) or in a bulk fashion after data input (ARCEDIT or INFO Moveitem & Calculate commands). One must be wary when initializing attributes during data input. The New command initializes all attributes of features to be added, to zero or blank, and the user must re-initialize all attributes to the desired values before entering features. Thus it is not possible to change the value of a single attribute without re-intializing all attribute values. The user must be particularly careful to ensure that the feature's unique identifier (User-id) is reset to its value before the New command is executed.

A facility also exists to delete unwanted attributes (Dropitem command).

6.2.1.5 Extract Features Based on Associated Attribute Values

ARC/INFO provides for the extraction of graphics features from coverages based on their attribute values. This can occur in two ways: 1) ARCEDIT can be used to select features (Editfeature & Select commands) and these can be written to a file (Put command), or 2) the ARC Reselect command accepts logical expressions and extracts features whose attributes match these expressions. The first approach is limited in that attributes associated with the graphics features are lost in the process. In the second approach, as with the appending of files together, only features of one type, e.g. arcs, may be extracted from a file at one time, but their associated attributes are preserved. The Reselect command can be used in conjunction with the facility to append files together (Adconc procedure) to extract any desired features, and subsequently combine them all into a single file integrating various feature types.

6.2.1.6 Determine Overlapping Features Between Two Files

There is no standard ARC/INFO command to determine the overlap between two files. A FORTRAN program (Ovrlap), with calls to ARC subroutines, was developed to determine overlap. Essentially, the software loops, reading each feature in the overlap file, and selects all features in the base file which fall within the area covered by the overlap feature (ARC Lbxsel routine). The software then does a feature by feature comparison for: 1) exact match, 2) partial match, or 3) no match. There is the option to delete the feature from the base and/or overlap files if an overlap occurs. If a match is partial, the feature is split and only the overlapping portion is deleted. A feature which matches exactly will always take precedence over a partial match. Between partial matches a function which determines the closeness of the match is used to determine which match will prevail in the event of two partial matches occurring in the base file for a given overlap feature.

6.2.1.7 Polygon Overlay

ARC/INFO provides extensive polygon processing facilities. The polygon overlay command (Identity command) operates on coverages containing points, lines, or polygons. Input consists of an input coverage and an overlay coverage of polygons. The command identifies features, or partial features, from the input coverage which fall within the polygon boundaries of the overlay coverage, appends the attributes associated with the overlay polygon, and writes the feature and attributes to an output coverage. To obtain a coverage which is a mix of points and lines, but not topologically built polygons, from the polygon overlay process the procedure must overlay points and lines separately and then append the files together (Adconc procedure).

6.2.2 Implementing and Testing the Approaches

The following sections describe how the four approaches to updating were implemented and tested, utilizing the ARC/INFO system.

Where appropriate, categories of the format used in the functional description (Section 5.2.2), have been utilized in describing implementations and tests.

The test data set consists of a cadastral map sheet in the Sutherland Shire of New South Wales. It contains approximately 500 arcs, 800 line segments, and 160 polygon labels (Figure 6.1). It covers an area of approximately 500 metres by 500 metres on the ground and was digitized at a scale of 1:1000. Tests were run using updates of varying number and spatial distribution. An update was seen as either a subdivision of one polygon into two or the amalgamation of two polygons into one. Each test was comprised of half subdivision updates and half amalgamation updates. In the figures, updates are represented as dotted lines. Five test cases were processed:

1) 1 update, representing 0.68% of the S72 file, (Figure 6.2)
2) 8 updates, representing 5% of the S73 file, in a sparse localized area, (Figure 6.3)
3) 8 updates, representing 5% of the S74 file, distributed throughout the file, (Figure 6.4)
4) 8 updates, representing 5% of the S75 file, in a dense localized area, (Figure 6.5)
5) 25 updates, representing 15% of the S76 file, a combination of all of the preceding updates (Figure 6.6).

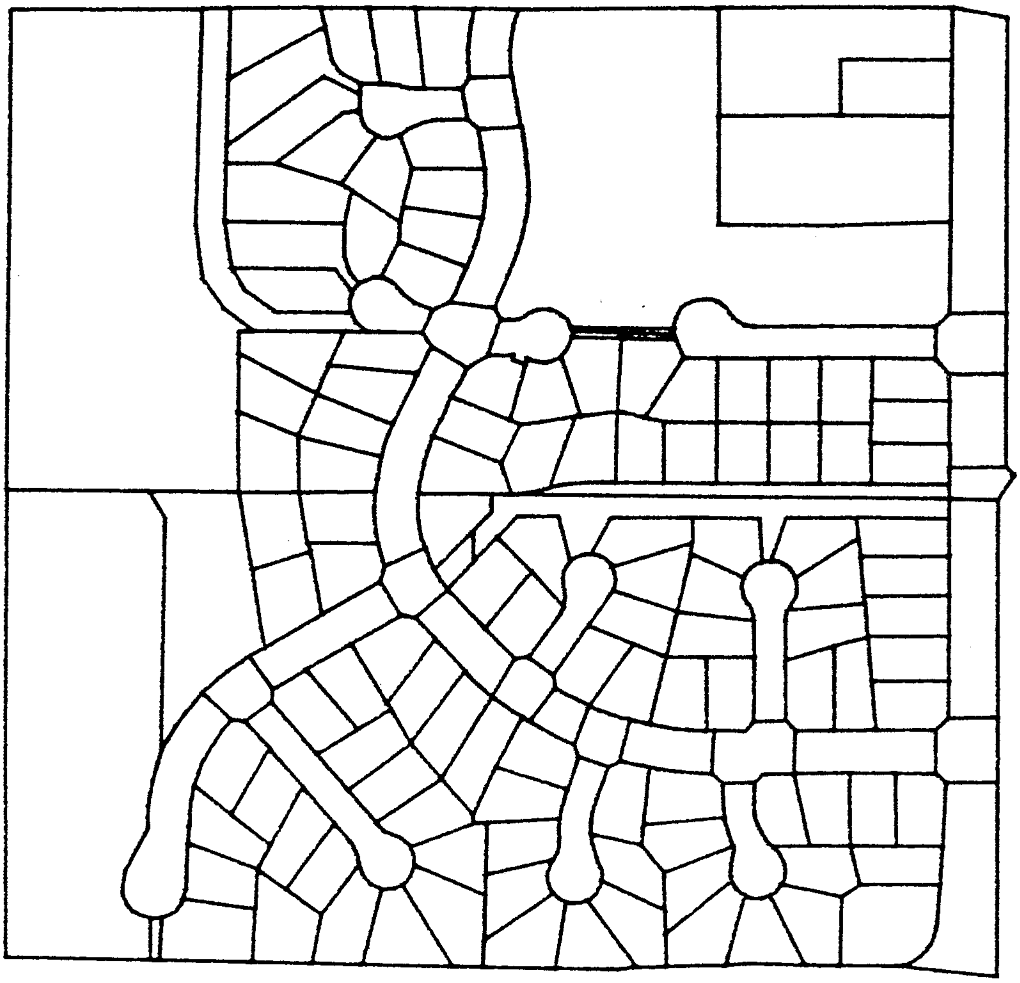


Figure 6.1 **Test Area in Sutherland Shire of New South Wales**

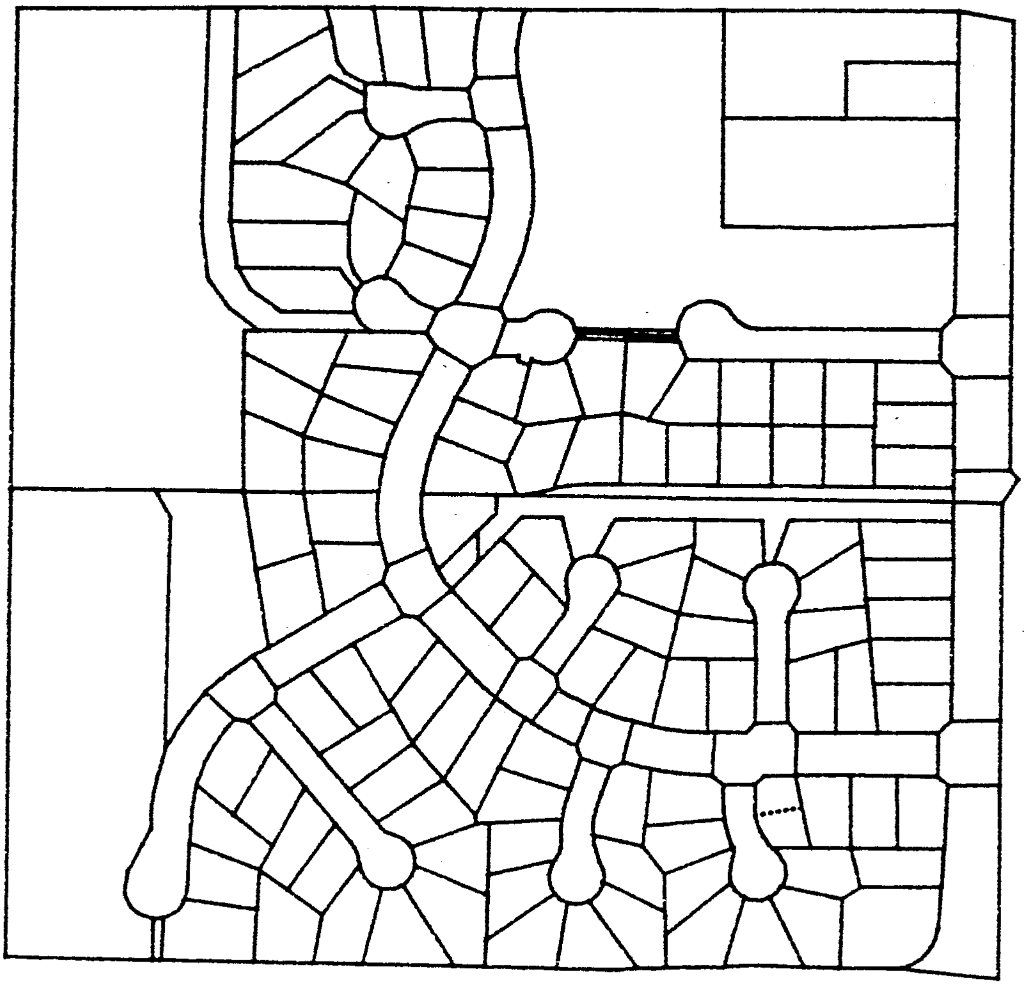


Figure 6.2 **One Update Representing 0.68% Of The S72 File**

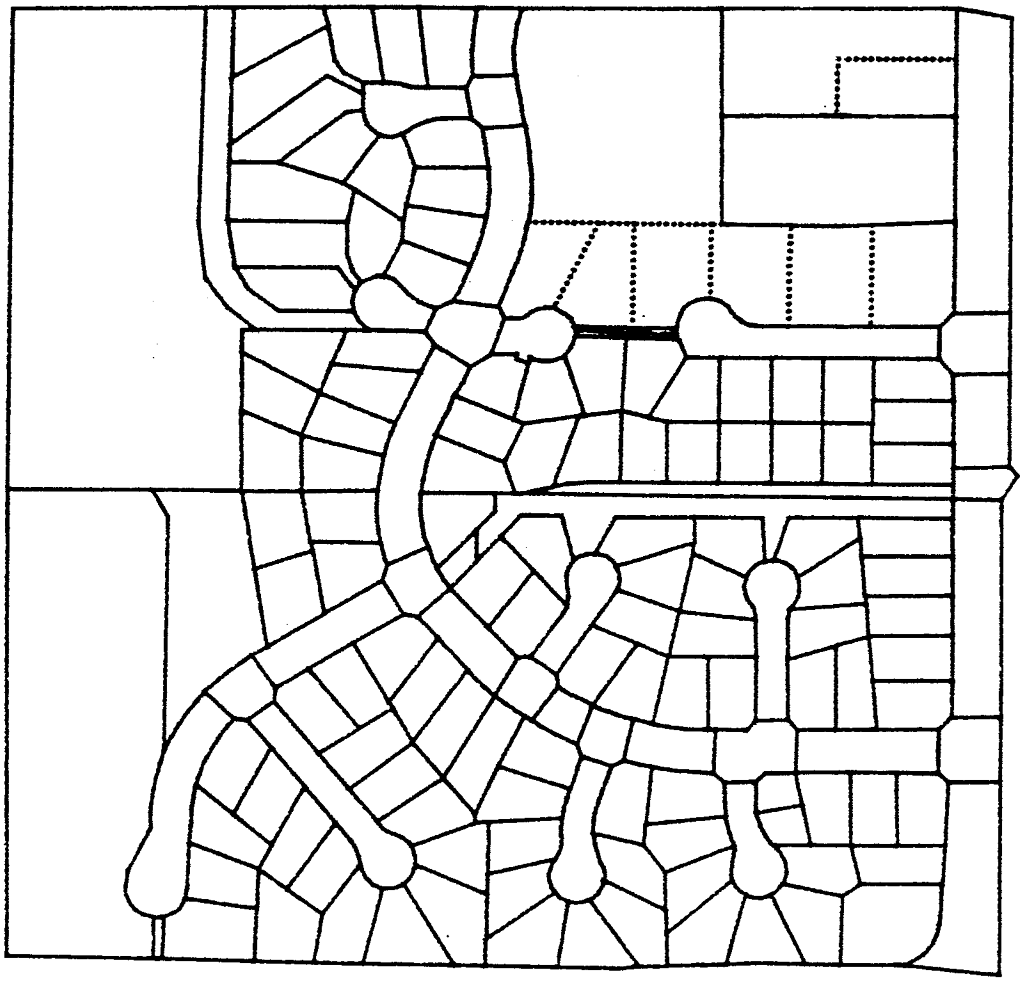


Figure 6.3 **Eight Updates, Representing 5% Of The S73 File, In A Sparse Localized Area**

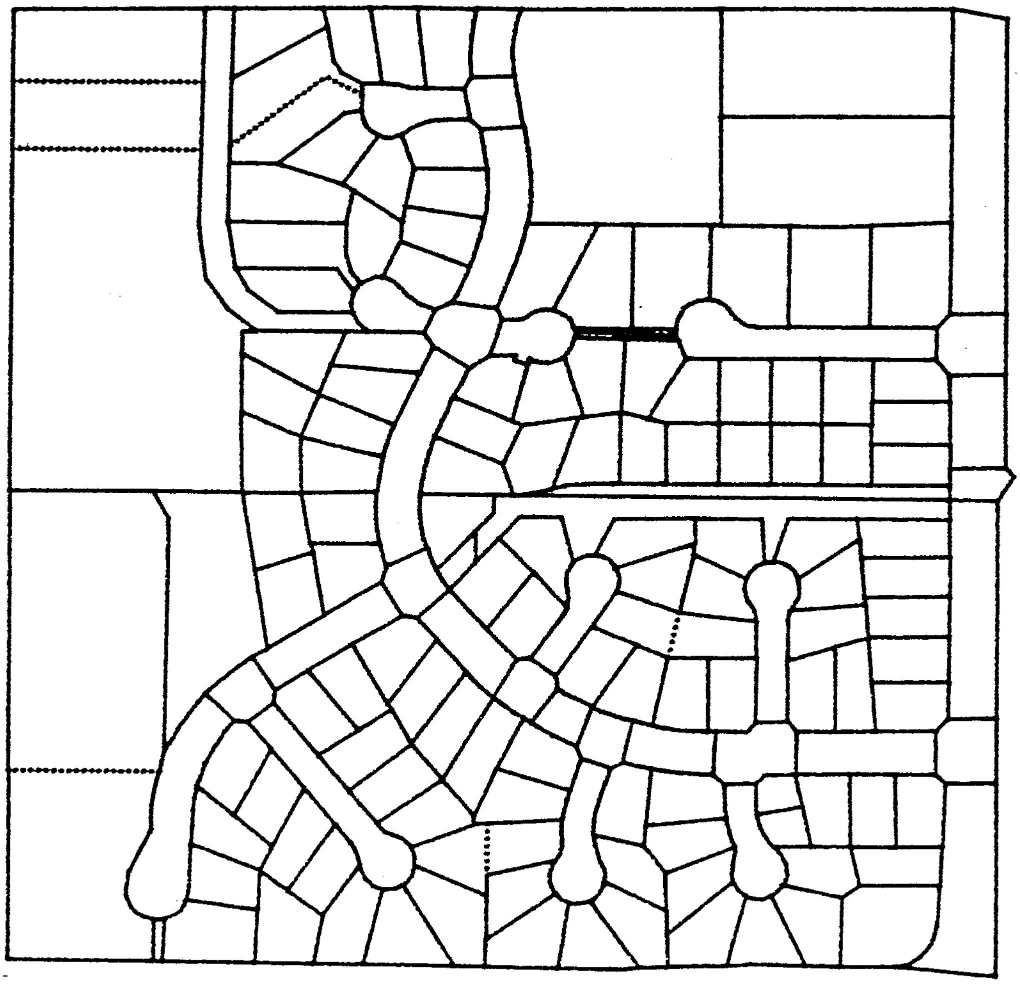


Figure 6.4 **Eight Updates, Representing 5% Of The S74 File, Distributed Throughout The File**

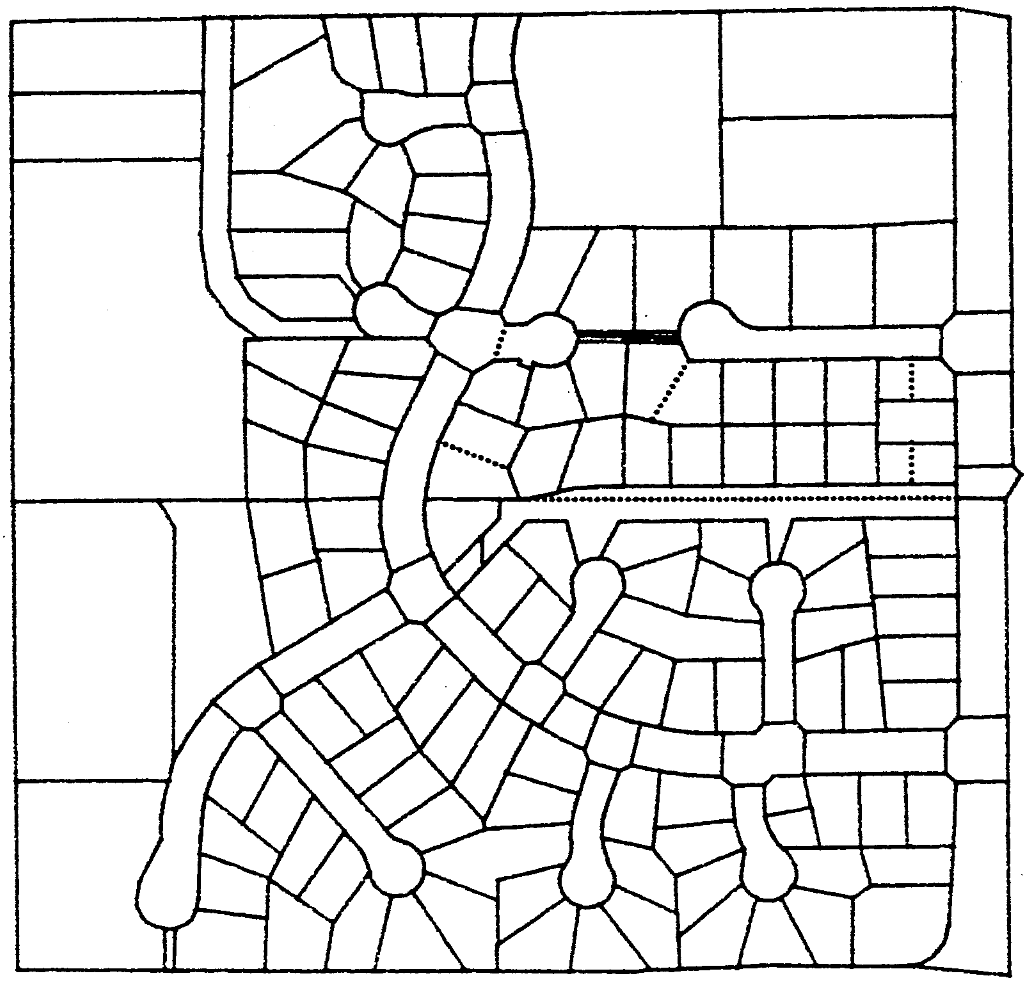


Figure 6.5 **Eight Updates, Representing 5% Of The S75 File, In A Dense Localized Area**

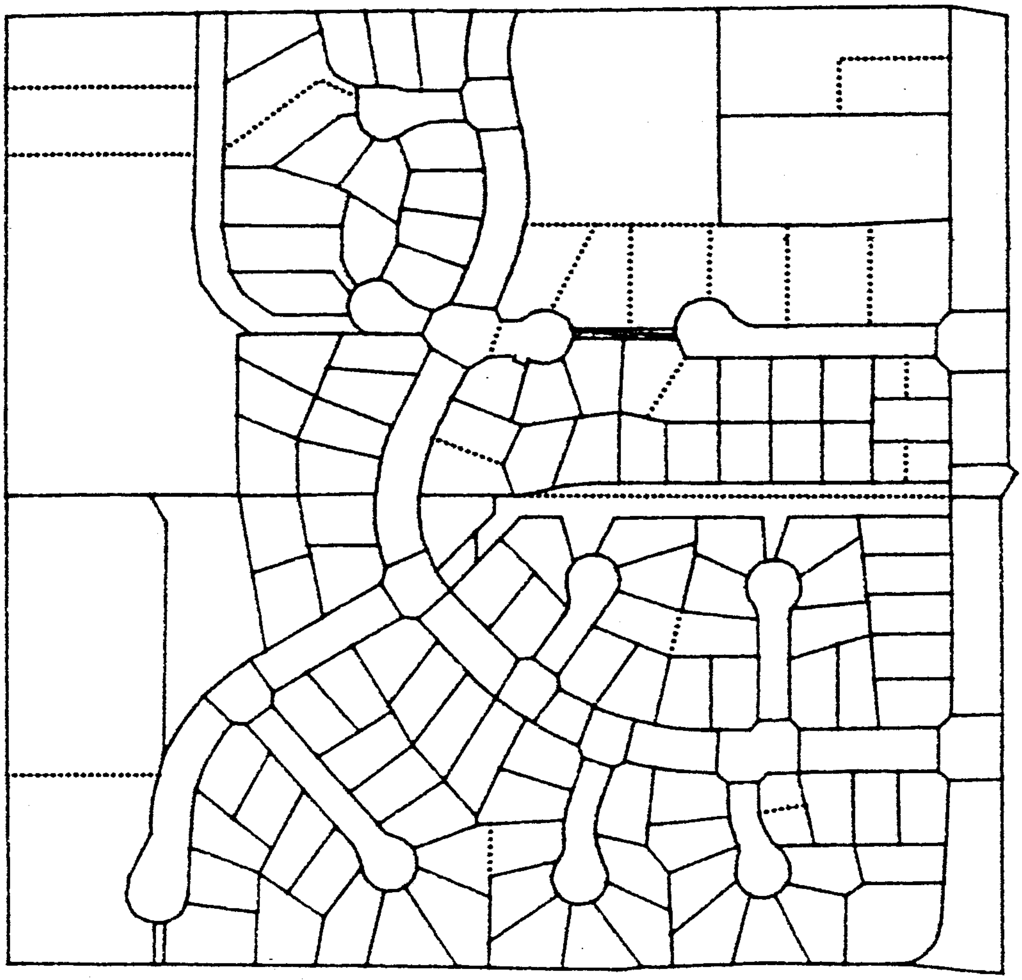


Figure 6.6 **Twenty-Five Updates Representing 15% Of The S76 File**

Each test was performed for all four approaches and in order to conserve time and disk space, the test files were created containing all attributes required by all approaches. A minimum of two attributes are required for incremental updating, a Graphic_Addition_Date and a Graphics_Supersession_Date (Section 5.2.2.4). These record when data is entered in the database and when it is superseded by more current data. Only one attribute is required for block updating, a Super_Block_Identifier (Section 5.2.2.2). This records which polygon in the Block Boundaries File that the feature falls within. In addition to these essential attributes, four optional attributes were included. They were a Feature_Code, to identify the feature; a Graphics_Data_Addition_Date, which records when data was collected; a Graphics_Data_Supersession_Date, which records when data is obsolete; and an Attribute_Modification field, to flag integrity constraints between graphics and attribute files. The feature code is fully implemented, the data addition and supersession dates are used to produce snapshots of the data as it existed on the ground at a point in time, and the attribute modification constraint has not been implemented as of yet. Additional attributes could easily be accommodated.

Tables 6.10 and 6.11 give the definition of the attributes associated with labels and arcs in the ARC/INFO files (see Section 6.1.1 for an explanation of INFO's field definition format). All fields before the Feature_Code (FC) are system generated.

The required attributes are defined by a procedure (Adatr) which utilizes ARC Additem commands. The attributes are initialize by another procedure (Adinit) which executes an INFO program to set the values of the fields. On the test data set this required 308 (5:08) cpu seconds and added 35 blocks (17.5 Kbytes) to the size of the file.

Due to the existence of additional attributes which are not required in the bulk and block update approaches, the bulk and block propagation updates will be slightly larger in disk size than if only the minimum attributes were stored. In the worst case the files could be as much as 15% larger. This magnitude of variance is not critical for tests between approaches and the relative file sizes will remain consistent within a given approach, enabling relative comparisons of the different test cases.

Table 6.10 **Attributes Associated With Labels in ARC/INFO.**

DATAFILE NAME: S60.PAT
11 ITEMS: STARTING IN POSITION 1

COL	ITEM NAME	WDTH	OPUT	TYP	N.DEC
1	AREA	4	12	F	3
5	PERIMETER	4	12	F	3
9	S60#	4	5	B	-
13	S60-ID	4	5	B	-
17	FC	4	4	C	-
21	GADD-D	8	8	D	-
29	GSUP-D	8	8	D	-
37	SUPBLK	4	4	C	-
41	GDAD-D	8	8	D	-
49	GDSP-D	8	8	D	-
57	ATRMOD	4	4	C	-

Table 6.11 **Attributes Associated With Arcs in ARC/INFO.**

DATAFILE NAME: S60.AAT
14 ITEMS: STARTING IN POSITION 1

COL	ITEM NAME	WDTH	OPUT	TYP	N.DEC
1	FNODE#	4	5	B	-
5	TNODE#	4	5	B	-
9	LPOLY#	4	5	B	-
13	RPOLY#	4	5	B	-
17	LENGTH	4	12	F	3
21	S60#	4	5	B	-
25	S60-ID	4	5	B	-
29	FC	4	4	C	-
33	GADD-D	8	8	D	-
41	GSUP-D	8	8	D	-
49	SUPBLK	4	4	C	-
53	GDAD-D	8	8	D	-
61	GDSP-D	8	8	D	-
69	ATRMOD	4	4	C	-

6.2.2.1 Bulk Updates

As outlined in Section 5.2.2.1, the bulk update approach requires no special structures to be established in the data file. The data file can be created and updated in any fashion. The sole requirement is that any time that the file is updated, the Last_Update_Date be recorded in the FILE_STATUS relation. In this prototype, the operator performing the update edit on the file is responsible for keying the new Last_Update_Date into the tracking database.

UPDATE FUNCTION	S72TIME	S72SIZE	S73TIME	S73SIZE	S74TIME
BULK EXTRACT	23	216	20	274	20
BULK EXPORT	37	216	46	274	45
BULK IMPORT	42	216	53	274	54
BULK APPLY	33	216	35	274	35
BULK CLEAN	106	216	107	274	104
TOTAL	241	216	261	274	258

UPDATE FUNCTION	S74SIZE	S75TIME	S75SIZE	S76TIME	S76SIZE
BULK EXTRACT	280	20	280	20	280
BULK EXPORT	280	44	280	44	280
BULK IMPORT	280	53	280	53	280
BULK APPLY	280	33	280	33	280
BULK CLEAN	280	103	280	103	280
TOTAL	280	253	280	253	280

UPDATE FUNCTION	AVGTIME	STDTIME	AVGSIZE	STDSIZE
BULK EXTRACT	20.60	1.34	266	28.07
BULK EXPORT	43.20	3.56	266	28.07
BULK IMPORT	51	5.05	266	28.07
BULK APPLY	33.80	1.10	266	28.07
BULK CLEAN	104.60	1.82	266	28.07
TOTAL	253.20	7.63	266	28.07

Table 6.12 **Results of Testing the Bulk Update Approach**

When an update is requested a copy of the file is made (extraction), it is reformatted to an agreed upon data interchange format (export), transferred, reformatted again to suit the user's system (import), the old copy is deleted on the destination system and the propagation update renamed to replace the destination data set (apply). Then if necessary, topological relationships are re-built for the data set (clean). This is implemented in ARC/INFO by using the ARC Copy command for extraction, ARC Export and Import commands for export and import, ARC Delete and Rename commands are combined in a procedure called Adubulk for the application, and the ARC Clean command for the clean. Update transfer considerations are left for the section on data transfer (Section 6.4).

Table 6.12 contains the results of testing the bulk update approach on the five cases. Avgtime and Avgsize give the average processing times and file sizes while Stdtime and Stdsize give the respective standard deviations. The table documents the processing times for each operation, in cpu seconds, and the data volume, in 512 byte blocks, of the propagation update. As would be expected the data volume is the entire file and remains relatively consistent, depending upon the balance between the additions and deletions constituting the update, along with the processing times. Note that the time to import data is consistently nine seconds longer than the time to export data and as such will be omitted from subsequent test results.

6.2.2.2 Block Update

Requirements - The block update approach requires the file to be partitioned into blocks. In the test data set, the file was divided into nine (9) blocks designated as A through I. The blocking was arbitrary but sought to follow the road boundaries, as closely as possible, and cover approximately equal areas (Figure 6.7).

File Creation - The file was created by digitizing all of the features. Once completed, the block boundaries were selected, written to a block boundaries file, and built into polygons. Then a polygon overlay was performed between the block boundary file and the complete data file. A procedure entitled Adbnit was then executed. It runs the ARC Identity command to overlay the block boundaries onto first the points, and then the lines, and to append the block boundary polygon's attributes to each feature inside a block boundary. The unnecessary block boundary polygon attributes are then stripped off (ARC Dropitem command) and the point and line files merged (Adconc procedure). The entire procedure requires 521 (8:41) cpu seconds. The other alternative is to enter the block identifiers during data entry. This approach tends to be time consuming and error prone.

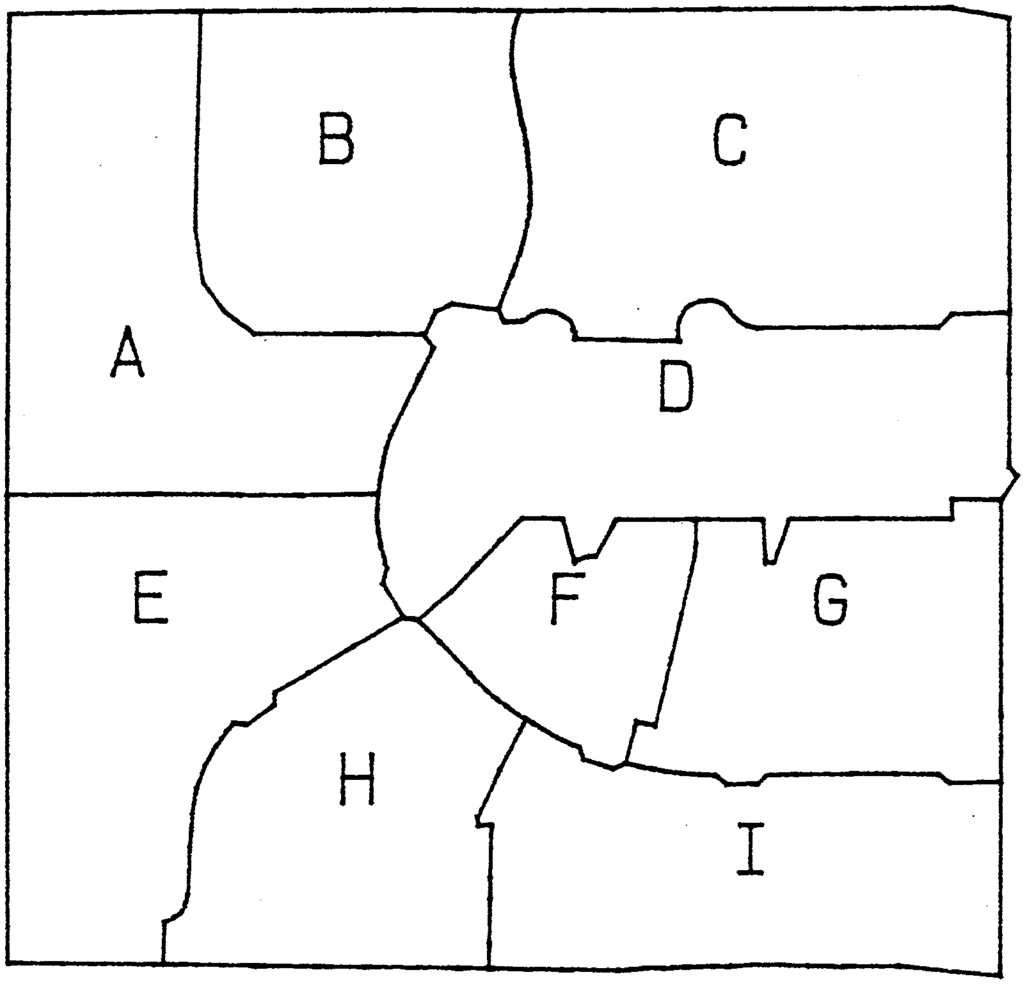


Figure 6.7 **Block Boundary File**

Recording - In the recording of source updates, each feature must be tagged with a block identifier and an entry must be made in the Block Update List of any block which is changed. The tagging of block identifiers was performed manually for the first three test cases along with the polygon overlay technique which was utilized for the file creation and all of the test cases. When the number of updates is small, it is a simple matter to perform the tagging manually but as the number of updates increases it is quicker and more accurate to perform it automatically.

A danger occurs when mixing automatic and manual tagging techniques. When features correspond with the boundary between two blocks some arbitrary but consistent rule must be applied for the assignment of the block identifier. It may be very difficult to ascertain on what bases a software program chooses which block identifier to assign to the feature on the border. It is of critical importance that the identification process be consistent.

The generation of the Block Update List is done on the basis of knowing the addition and supersession dates, along with the block identifier. Normally, the edit software would be required to write out the identifier of the block being modified to the block update list but since incremental update attributes were already stored, it was simpler to use this approach than to modify the edit software. A procedure (Adibul) runs an INFO program which selects all features which have been modified since a desired date, i.e. have an addition or supersession data more recent than the desired date but not equal to infinity, and writes the block identifier and date of update to the Block Update List. Finally, duplicate entries are eliminated. As is seen in Table 6.13, the time to generate a Block Update List is very small, on average 16 cpu seconds. The file itself is only 1 or 2 blocks in size.

Extracting - Updated blocks are extracted via the Adeblk procedure. It is a command language procedure which accesses the Block Update List and determines which blocks have been updated since a particular date. Then ARC Reselect commands are executed to extract the features in these blocks and finally merged together (Adconc procedure) into a single file. As would be expected, Table 6.13 illustrates that the size of the propagation update is directly related to the spatial distribution of the updates relative to the size of a block. The time to extract the blocks is relatively stable and is independent of the volume of data extracted from the file.

Application - The application of updates is performed by the Adublk procedure. This involves determining which blocks occur in the propagation update (Adbbul procedure), deleting these features from the destination file (ARCEDIT batch command file), and merging the propagation updates in (Adconc procedure). As Table 6.13 shows, the time to apply block updates is stable and independent of the size of the propagation updates.

UPDATE FUNCTION	S72TIME	S72SIZE	S73TIME	S73SIZE	S74TIME
BLOCK BLKINT	529	202	522	209	536
BLOCK NUMBLK		1		1	
BLOCK BLKBLD	16	1	15	1	16
BLOCK EXTRACT	206	34	204	19	216
BLOCK EXPORT	17	34	14	19	33
BLOCK IMPORT	26	34	23	19	42
BLOCK APPLY	269	34	210	19	232
BLOCK CLEAN	104	265	106	274	101
TOTAL	638	34	572	19	640

UPDATE FUNCTION	S74SIZE	S75TIME	S75SIZE	S76TIME	S76SIZE
BLOCK BLKINT	211	536	210	536	211
BLOCK NUMBLK	7		1		9
BLOCK BLKBLD	2	16	1	16	2
BLOCK EXTRACT	160	205	43	221	210
BLOCK EXPORT	160	18	43	38	210
BLOCK IMPORT	160	27	43	47	210
BLOCK APPLY	160	197	43	238	210
BLOCK CLEAN	283	100	278	102	278
TOTAL	160	563	43	662	210

UPDATE FUNCTION	AVGTIME	STDTIME	AVGSIZE	STDSIZE
BLOCK BLKINT	531.80	6.26	208.60	3.78
BLOCK NUMBLK			3.80	3.90
BLOCK BLKBLD	15.80	.45	1.40	.55
BLOCK EXTRACT	210.40	7.64	93.20	86.07
BLOCK EXPORT	24	10.75	93.20	86.07
BLOCK IMPORT	33	10.75	93.20	86.07
BLOCK APPLY	229.20	27.73	93.20	86.07
BLOCK CLEAN	102.60	2.41	275.60	6.73
TOTAL	615	44.49	93.20	86.07

Table 6.13 **Results of Testing the Block update Approach**

6.2.2.3 Incremental Update Without Attached Attributes

As outlined in Section 5.2.2.3, the incremental update without attributes approach requires no special structures to be established in the data file. The data file can be created and updated in any fashion. The only requirement is that a copy of the file before the update be retained.

UPDATE FUNCTION	S72TIME	S72SIZE	S73TIME	S73SIZE	S74TIME
INCWOA EXTRACT	823	10	883	13	928
INCWOA EXPORT	25	10	26	13	26
INCWOA IMPORT	34	10	35	13	35
INCWOA APPLY	172	10	185	13	181
INCWOA CLEAN	105	265	103	274	102
TOTAL	1159	10	1232	13	1272

UPDATE FUNCTION	S74SIZE	S75TIME	S75SIZE	S76TIME	S76SIZE
INCWOA EXTRACT	15	905	15	916	28
INCWOA EXPORT	15	26	15	27	28
INCWOA IMPORT	15	35	15	36	28
INCWOA APPLY	15	183	15	194	28
INCWOA CLEAN	282	104	278	103	277
TOTAL	15	1253	15	1276	28

UPDATE FUNCTION	AVGTIME	STDTIME	AVGSIZE	STDSIZE
INCWOA EXTRACT	891	41.47	16.20	6.91
INCWOA EXPORT	26	.71	16.20	6.91
INCWOA IMPORT	35	.71	16.20	6.91
INCWOA APPLY	183	7.91	16.20	6.91
INCWOA CLEAN	103.40	1.14	275.20	6.38
TOTAL	1238.40	47.70	16.20	6.91

Table 6.14 **Results of Testing the Incremental Update Without Attributes Approach**

Extracting - Updates are extracted by determining the overlap between the before and after update files, and deleting the overlapping features. The remaining features in the original file are the supersessions and in the updated file the additions. Thus two files are created, one containing supersession updates and the other containing addition updates. This was implemented in two ways: 1) creating ARCEDIT command procedures to graphically overlay the two files, highlight the additions and supersessions, and manually extract the updates (Adedt procedure), or 2) determine the updated features automatically (Addinc procedure utilizing Ovrlap, Section 6.2.1.6). As recorded in Table 6.14, the automated detection of changes between two files is processing intensive. The primary advantage is that the resulting propagation updates are small in size relative to other update approaches.

Application - Propagation updates are applied with the Aduinc procedure. It uses the Ovrlap facility (Section 6.2.1.6) to delete features in the destination file which correspond to supersession features in the supersession file and then appends features from the additions file (Adconc procedure). Table 6.14 appears to show that the application of updates in this fashion is stable, with respect to processing time, and independent of the number of updates applied. The range in the number of updates tested is really too small to draw any conclusion of this nature.

6.2.2.4 Incremental Update With Attached Attributes

Requirements - Incremental updates with attached attributes require that an addition and supersession date be associated with each feature in the file.

File Creation - In the creation of the file, data was digitized and when input was complete attributes were defined (Adatr procedure) and initialized (Adinit procedure). This took 308 cpu seconds and resulted in a file of 251 blocks. An attempt was also made to assign attributes as the data input was occurring (Addef and Adadd procedures), but this was awkward and error prone as compared to initializing attributes after data input.

Care must be taken when topologically cleaning ARC/INFO files with attached attributes. If due to a digitizing error, e.g. a missing arc, two labels occur in one polygon only the first label encountered will be stored as the polygon record. The other label will remain in the graphics file but its attributes will be stripped off. Thus in order to retain attributes associated with labels, the uncleaned file must be edited, and not the cleaned file.

UPDATE FUNCTION	S72TIME	S72SIZE	S73TIME	S73SIZE	S74TIME
INCWA EXTRACT	426	10	376	13	423
INCWA EXPORT	25	10	26	13	26
INCWA IMPORT	34	10	35	13	35
INCWA APPLY	172	10	185	13	181
INCWA CLEAN	105	265	103	274	102
TOTAL	762	10	725	13	767

UPDATE FUNCTION	S74SIZE	S75TIME	S75SIZE	S76TIME	S76SIZE
INCWA EXTRACT	15	425	15	420	28
INCWA EXPORT	15	26	15	27	28
INCWA IMPORT	15	35	15	36	28
INCWA APPLY	15	183	15	194	28
INCWA CLEAN	282	104	278	103	277
TOTAL	15	773	15	780	28

UPDATE FUNCTION	AVGTIME	STDTIME	AVGSIZE	STDSIZE
INCWA EXTRACT	414	21.37	16.20	6.91
INCWA EXPORT	26	.71	16.20	6.91
INCWA IMPORT	35	.71	16.20	6.91
INCWA APPLY	183	7.91	16.20	6.91
INCWA CLEAN	103.40	1.14	275.20	6.38
TOTAL	761.40	21.43	16.20	6.91

Table 6.15 **Results of Testing the Incremental Update With Attributes Approach**

Recording - During source updating, the addition and supersession dates must be set for each edited feature. The Adadd procedure builds ARCEDIT command procedures to assign these attributes. When an addition is to be made the procedure to add a feature is activated, via keyboard or menu input. It will initialize the required attributes and set the environment for the user to digitize the features. Similarly for supersessions, the user selects the features to be superseded and a command procedure is executed to set the supersession date.

Extracting - The Adres2 procedure selects all features in a coverage which have an addition date greater than the desired date and a supersession date equal to infinity, and all features which have a supersession date greater than the desired date and not equal to infinity. The additions and supersessions are stored in separate files. Table 6.15 shows the times required for extracting incremental updates and the resulting file sizes. The time required to extract the updates is approximately one half of the time required to detect incremental updates without attributes. Thus the overhead in storing addition and supersession dates bears fruit in reducing the time to extract the updates.

Application - The procedure to apply incremental updates with attributes is identical to that for incremental updates without attributes (Section 6.2.2.3).

6.2.2.5 Alerts

As outlined in Section 5.2.2.5, the alert update approach requires no special structures to be established in the data file. The data file can be created and updated in any fashion. The only requirement is that in order for update markers to be generated some means of detecting updates must exist. Detecting updates can be performed in the following ways: 1) using the Adedt procedure to graphically overlay the files highlighting the updates, 2) deleting overlapping features between files before and after update (Addinc procedure, Section 6.2.2.3), or 3) selecting updates based on attributes (Adres2 procedure, Section 6.2.2.4). Once updates are detected then alerts can be manually added or automatically generated. The results of automatically generating alerts (Adialtf procedure) from incremental updates are given in Table 6.16. The approach minimizes processing time and produces small propagation update files. The application of the updates is performed by graphically overlaying the alert file with the destination data (Section 6.2.1.1).

UPDATE FUNCTION	S72TIME	S72SIZE	S73TIME	S73SIZE	S74TIME
ALERT GENERATE	90	3	91	5	90
ALERT EXPORT	12	3	12	5	12
ALERT IMPORT	21	3	21	5	21
TOTAL	123	3	124	5	123

UPDATE FUNCTION	S74SIZE	S75TIME	S75SIZE	S76TIME	S76SIZE
ALERT GENERATE	5	91	6	91	12
ALERT EXPORT	5	12	6	13	12
ALERT IMPORT	5	21	6	22	12
TOTAL	5	124	6	126	12

UPDATE FUNCTION	AVGTIME	STDTIME	AVGSIZE	STDSIZE
ALERT GENERATE	90.60	.55	6.20	3.42
ALERT EXPORT	12.20	.45	6.20	3.42
ALERT IMPORT	21.20	.45	6.20	3.42
TOTAL	124	1.22	6.20	3.42

Table 6.16 **Results of Testing the Alert Update Approach**

6.3 Reformatting Updates

Reformatting of updates in the prototype utilized the ARC/INFO facilities to generate ARC/INFO interchange format and Digital Line Graph (DLG) format files for graphics and INFO's facilities to produce ASCII, comma delimited, dBase, Lotus, and other attribute formats. The link between the graphics and attribute files was not lost in the reformatting process. Time and effort are expended when the link between the graphics and attribute files is lost during data reformatting and must be restored via manual intervention.

6.4 Transferring Updates

The prototype system was implemented on a single computer system. Therefore, no data transfer issues arose. Files were simply copied from one user to another either within the system or to other computers over an Ethernet communications network.

6.5 Verifying Updates

The prototype system verifies updates in two ways: 1) comparison to source, and 2) logical consistency through topological tests. The first approach utilizes the same techniques as detecting incremental updates in Section 6.2.2.3. It can be performed graphically with the Adedt procedure or automatically with the Addinc procedure. The times quoted in Table 6.14 for the Addinc procedure are the same as for verifying files. The difficulties with this approach are that: 1) the user must have the updated file and a copy of the source file, and 2) the automated detection process is very cpu intensive.

Using topological consistency tests is very rigorous, timely (average of 105 cpu seconds for a topological check versus 891 cpu seconds for an automated overlap check), and does not require a source file for comparison. The topological test is ideal, if the facilities exist and the data is topologically structured, but a comparison to source should be performed on a periodic basis to determine if there is a problem with the update processor.

6.6 Summary

The ARC/INFO system provided open ended facilities to prototype the data update process. The testing of approaches to updating produced valuable insights into the strengths and limitations of the approaches. This information will be used as the basis for developing a criteria for the selection of an optimum update policy (Section 7.2).

Chapter 7 CRITERIA FOR THE SELECTION OF AN OPTIMUM UPDATE POLICY

The following sections describe general criteria for the selection of an optimum approach to data updating. With all of the criteria presented, it must be understood that the participants in the land information network must determine and define the factors to be optimized. The following sections will assist a system manager in determining which approach will suit his particular system's requirements, but in the final analysis the update policy will be forged out of the requirements of a number of groups. Thus, an attitude of co-operation and compromise must be adopted if land information system managers are to work together and see their land information network move from merely exchanging data to updating exchanged data.

The criteria presented have been developed on the basis of the prototype described in Chapter 6. It must be emphasised that in determining optimum update policies a wide range of factors must be considered. For example, the characteristics of the data in the database, the characteristics of the updates, system capabilities, ease of expanding the system capabilities, requirements for timely updates, and the budget allocated to support the updating system all enter into the decision making process. The number and variability of these factors makes it impossible to define specific criteria for determining optimum update policy. Therefore general criteria are presented and the system manager must use these to reduce the multitude of options to a few viable alternatives which can be specifically investigated in light of the specific requirements of each approach, as described in Chapter 5.

Given the difficulties in determining measures of an optimum approach to updating, especially on the basis of one implementation, care must be exercised in the application of these criteria. These criteria would benefit immensely from further testing. In cases where the test results, from this research, have not agreed with the expected results, an explanation is given.

There are two fundamental perspectives from which a suitable approach to data updating may be determined: 1) ascertain which approaches are possible, with the current system capabilities, and select the most appropriate of these, or 2) ascertain the level of performance required for the data update process, select an approach which will produce the desired results, and add any required system capabilities to achieve the required level of performance. Obviously the first perspective seeks to process updates with no additional system development and without regard for update processing performance. The second perspective is concerned only with achieving the required update processing performance and will incur any system development cost to achieve this. In reality both of these approaches exist, e.g. an agency with a small budget in the land information systems area versus an agency with a statutory responsibility to provide up-to-date land information, but normally capability and performance will each be compromised on the basis of costs and benefits. Therefore, the criteria for the selection of an optimum update policy, in this thesis, will be dealt with according to the following three categories: 1) capability, 2) performance, and 3) cost/benefit.

7.1 Capability Criteria

The capabilities required, in a land information system, to support the data update approaches must be examined. The user must determine which approaches to data updating are supported by the tools available on the user's land information system. The requirements for each approach are detailed in Chapter 5. The following tables summarize the capabilities required for each update facility. In most instances a variety of levels of sophistication can be employed to provide the required capability.

The left most entry in the table describes the function to be performed and subsequent entries define the required capabilities to support this function. Indented entries provide alternative means of satisfying the requirement, under which they are indented. For a land information system to be capable of performing a function, it must be capable of performing at least one of the requirements indented under the function. For example, from Table 7.1 it can be seen that a facility to track updates can be implemented in three ways:

1) conventional card file,
2) computerized database management system utilizing manual input, or
3) computerized database management system utilizing automated input and a digital communications network.

Table 7.1 **Requirements for Tracking Updates** (Section 5.1)
Conventional card file. Computerized database management system. -Manual input. -Automated input. -Digital communications network.

Table 7.2 **Requirements for Bulk Updating** (Section 5.2.2.1)
Record that an update has occurred in the tracking database. -Manually. -Automatically. Extract and apply updates. -Duplicate a file. -Delete a file.

Table 7.3 **Requirements for Block Updating** (Section 5.2.2.2)
Initialize and maintain block identifiers. -Manually during data input. -Automatically via polygon overlay. Maintain the block update list. -Manually after each edit session. -Automatically via the edit software. -Comparison of before and after files. -Manually deleting overlapping features. -Polygon overlay. -Automatically deleting overlapping features. -Polygon overlay. -Derive from incremental updating information. Extract blocks. -Select features with a given attribute and write them to a file. Apply blocks. -Select features with a given attribute and delete them. -Append two files together.

Table 7.4 **Requirements for Incremental Updating Without Attached Attributes** (Section 5.2.2.3)
Retain a copy of the source file prior to updating. Extract addition and supersession updates. -Manually delete overlapping features. -Automatically delete overlapping features. Apply addition and supersession updates. -Append two files together. -Manually delete overlapping features. -Automatically delete overlapping features.

Table 7.5 **Requirements for Incremental Updating With Attached Attributes** (Section 5.2.2.4)
Initialize and maintain addition and supersession date attributes. -Manually during data input. -Batch process after data input. Extract updates. -Select features with a given attribute and write them to a file. Apply updates. -Append two files together. -Manually delete overlapping features. -Automatically delete overlapping features.

Table 7.6 **Requirements for Alert Updates** (Section 5.2.2.5)
Record updates in the same manner as for a bulk, block, or incremental approach. Generate alerts. -Extract updates in the same manner as for a bulk, block, or incremental approach. -Create markers representing the extracted updates. -Manually via edit facilities. -Automatically via software. Apply alerts. -Graphically overlaying two files. -Appending two files together.

By comparing the capabilities required by a particular update facility and the capabilities which exist in a land information system it is possible to determine which methods of updating can be implemented without the development of additional tools.

7.2 Performance Criteria

The performance of various approaches to updating in a land information network has been tested (Section 6.2.2). This section seeks to compare the various approaches on the following basis:

1) Complexity of update processing;
2) Volume of propagation updates;
3) Spatial distribution and number of updates; and
4) Timeliness requirements.

Table 7.7 contains the average values and standard deviations for the processing time (cpu seconds) and file size (512 byte blocks) tests performed. These values are taken from Tables 6.12 to 6.16.

Table 7.7 **Average Values for Processing Time and File Size Tests**

Approach	Time Average	Time Std Dev	Size Average	Size Std Dev
Bulk	253.2	7.6	266.0	28.0
Block	615.0	44.5	93.2	86.0
Incremental without attributes	1238.4	47.7	16.2	6.9
Incremental with attributes	761.4	21.4	16.2	6.9
Alerts	124.0	1.2	6.2	3.4

7.2.1 Complexity of Update Processing

Figure 7.1 plots the average processing time, in cpu seconds, for each approach. The standard deviations of the processing times are indicated by the shaded boxes above and below the average processing time. Note that the transfer time is not included in these figures. If it is assumed that the processing time for alerts is 1 unit, the approximate relative processing times for these tests would be: bulk - 2x, block - 5x, incremental with attributes - 6x, and incremental without attributes - 10x. It is important to note that the processing times increase as the complexity of the processing increases, and not as a result of increasing the volume of data to be processed. This yields guidelines in the following areas:

1) **Manual intervention:** If a user wishes to perform a particular update approach via manual editing, rather than software, the relative amount of manual effort expended for each approach will be proportional to the relative processing times.

2) **System sophistication:** If a user wishes to perform a particular update approach via an automated software solution, the level of sophistication required of the software will be proportional to the relative processing times.

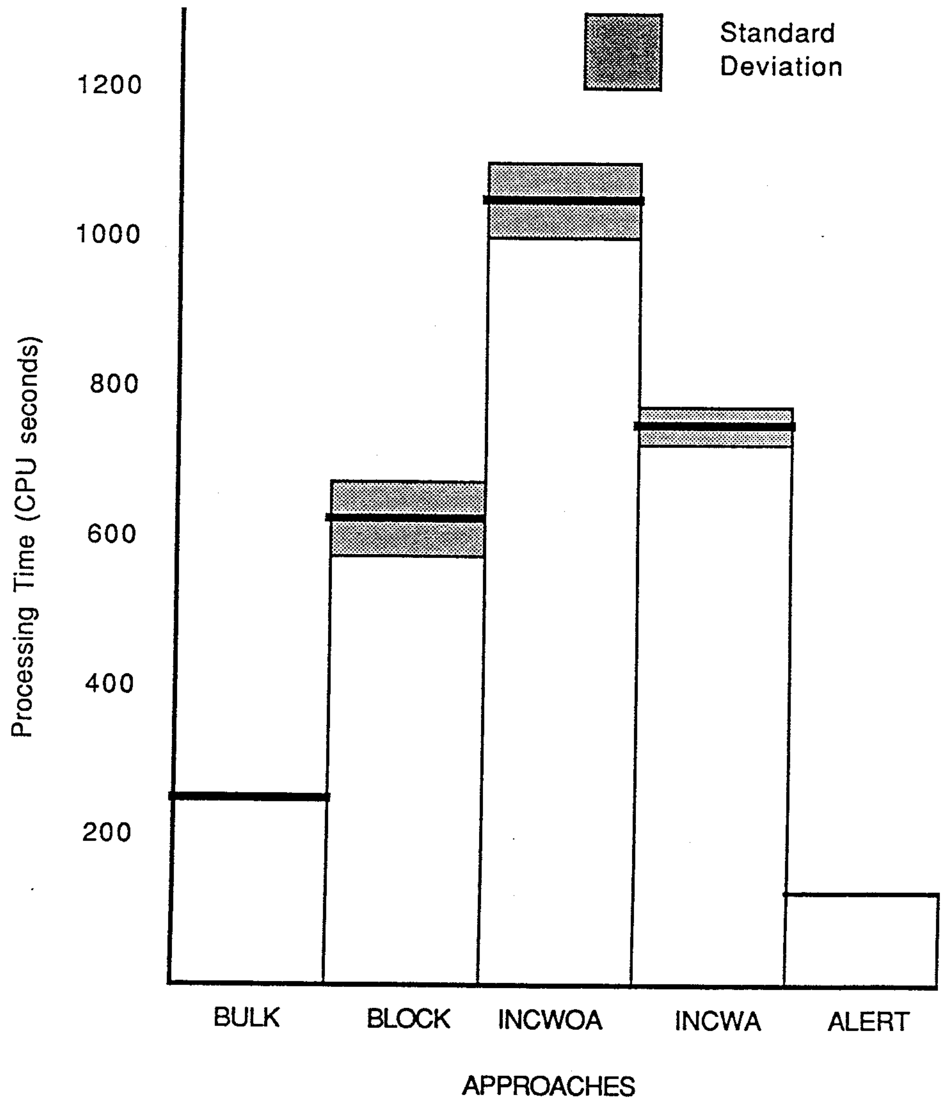


Figure 7.1 **Average Processing Times for Tests of Update Approaches**

3) **Probability of error:** The probability of an error occurring in a particular updating approach is proportional to the complexity of the approach. Thus, the possibility of error will increase in a similar fashion to the relative processing times. The relative complexity of the approaches is evident in the sophistication required in the following areas: 1) manual editing operations, 2) software tools, 3) data structures. As the level of sophistication increases, it is more and more difficult to develop procedures and software which will accommodate every possible special case. Even though the update server is designed and developed to be robust, the likelihood of an unforeseen problem occurring increases with the complexity of the system.

4) **Processing efficiency:** The amount of processing time required for data updating can be very significant (e.g. 20 cpu minutes for incremental updates without attributes) and the differences between approaches are large. If the level of computing resources available to the update process is limited, then the less complex approaches will be most favourable.

7.2.2 Volume of Propagation Updates

Figure 7.2 plots the average file size of the propagation updates, in 512 byte blocks, for each approach. The standard deviations of the file sizes are indicated by the shaded boxes above and below the average size.

If it is assumed that the size of the alert propagation updates is 1 unit, the approximate relative file sizes would be: incremental without attributes - 3x, incremental with attributes - 3x, block - 15x, and bulk - 44x. The large standard deviation associated with the block update approach will be dealt with in Section 7.2.3. The volume of the propagation updates generated by an update approach will have a significant impact on the following areas:

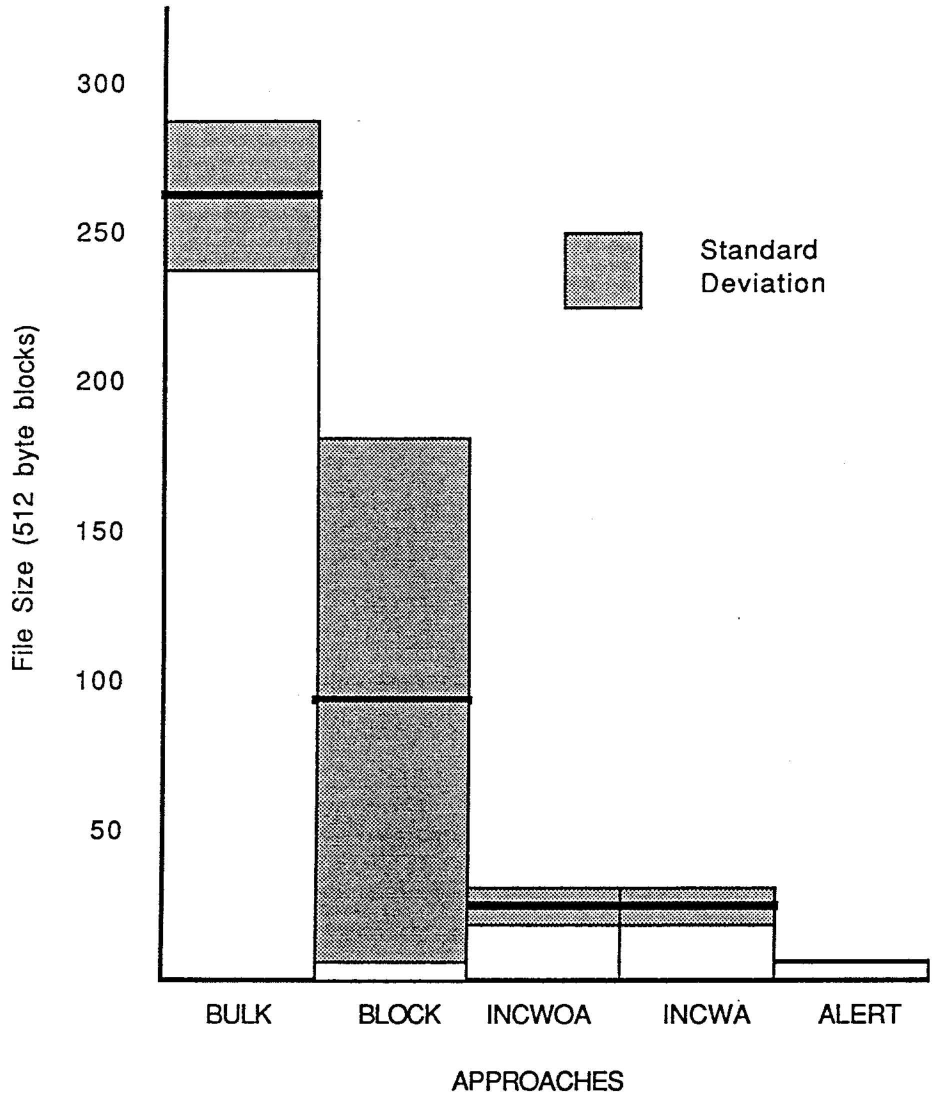


Figure 7.2 **Average File Sizes for Tests of Update Approaches**

1) **Transfer time for digital communication:** The transfer time over a digital communications network is directly proportional to the volume of data to be transferred. Transferring data at a rate of 4800 bits/second would require 6.6 seconds to transfer the 6.2 blocks (3.1 Kbytes) required for the average alert update. The bulk update's 266 blocks (136 Kbytes) would require 4.7 minutes. A typical 1:50000 scale topographic map file of 15 Megabytes (COVER et al., 1983) would require 8.7 hours. It would probably be more cost effective to send 15 Megabyte files on magnetic media via a courier service than over a 4800 bits/second communications line. Thus if updates are to be transferred on-line, approaches which minimize the volume of data are appropriate. Current advances in communications technology are rapidly increasing the speed of data transfer and thus reducing the concern over the volume of data to be transferred. Nonetheless, as the speed of a communication network increases so does the cost.

   Alternatively, if data is to be transferred via magnetic media, the only portion of the transfer time which is dependent upon the volume of the data is the actual recording of the data on the media. In recording, a significant portion of the time is spent in setting up the media and all but very large variations in data volumes will go unnoticed.

2) **Effort to re-establish topological relationships:** If the data being updated has topological relationships built, the updates will affect the topological structure of the file and require that the topology be re-built. Many land information systems have the capacity to re-build topology for only the updates and surrounding features, rather than re-building the relationships for the entire file. Thus, minimizing the volume of changes can reap extensive processing time savings in the re-establishment of topological relationships in the updated file.

   ARC/INFO provides facilities for local, or incremental, topological processing but the average processing time, in all of the update approaches, for re-building topology after an update is 103 +/-2 cpu seconds. The low standard deviation indicates that there is no significant variation in the time to build topology across the different updating approaches. This is due to the fact that ARC/INFO version 3.2.1 stores the attributes for points and polygons in the same file. When point updates are to be applied to a file, the topological relationships must be stripped off to convert the polygon attribute file to a point attribute file and once the updates are applied, the polygon attributes are re-built. Thus the topology must be re-built for the entire file even if only a single update is applied to the file. It is expected that in other systems which support local topological building, and in ARC/INFO version 4.0, minimizing the volume of updates will speed up the re-establishment of topological relationships.

3) **Effort to Re-associate Attributes and Graphics:** If during the data reformatting and transfer, the link between the attribute and graphics files is lost and must be re-established manually (Section 5.3), the amount of effort which must be expended is directly proportional to the volume of data which has been affected.

7.2.3 Number and Spatial Distribution of Updates

The effectiveness of a particular data update approach can be dramatically affected by the number and spatial distribution of the updates. This is evident from the large standard deviations associated with particular updating approaches recorded in Table 7.7. As would be expected, the volume of bulk propagation updates is stable. For each update request the entire file is transferred. The volume of block updates varies widely. Essentially, if the updates are localized then a small number of blocks will be affected but if the updates, even if small in number, are widely distributed a large number of blocks will be affected. The volume of incremental and alert updates is directly proportional to the number of updates which have occurred, independent of their spatial distribution.

Figure 7.3 graphs the most suitable approaches for data updating based on the number and spatial distribution of updates.

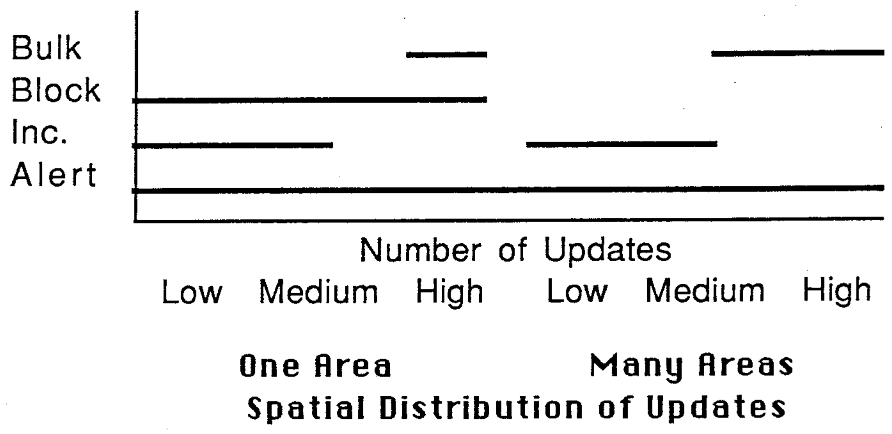


Figure 7.3 **Optimum Update Approaches Based on the Number and Spatial Distribution of Updates**

7.2.4 Timeliness Requirements

The capacity for timely updates is of major concern to users in a land information network. It is not possible, in general terms, to specify whether particular update approaches will satisfy a particular timeliness requirement. The time taken to process updates is dependent upon the frequency, number, volume, and spatial distribution of updates, the efficiency of the implementation of the data update model, and the method of data transfer utilized. Thus a user must have a good understanding of the characteristics of the updates being propagated and the system being used in order to properly estimate the capacity of an update approach to service updates in a timely fashion.

Based upon the experience gained in this research, the assumption that updates are occurring regularly, e.g. daily, and the assumption that the data involved is "normal" with respect to volume and density, a subjective scale of appropriate approaches versus timeliness requirements has been developed. Figure 7.4 graphs the most suitable approaches for data updating based on varying requirements for timely updates. It must be noted that this scale is entirely subjective and serves merely as a starting point in considering the ability of a particular data updating approach to satisfy a requirement for timely updates.

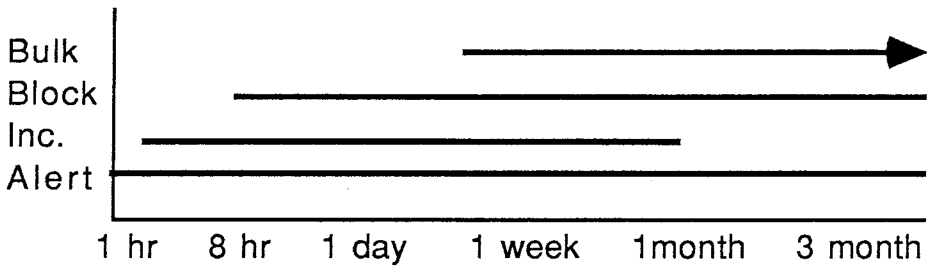


Figure 7.4 **Optimum Update Approaches Based on Timeliness Requirements**

7.3 Cost/Benefit Criteria

The choice of appropriate data updating techniques for source and destination systems in a land information network will necessitate a trade-off between existing capabilities and desired performance on the basis of their respective costs and benefits. The following discussion is intended to highlight some of the factors which must be considered in comparing the costs and benefits of the various updating techniques. It is by no means an introduction to cost/benefit analysis.

Weir and Swetnam [1986] outline a technique to assess the economic benefits of land information system projects. In this technique, benefits are essentially of two types: 1) production, and 2) consumption. Production benefits are those which result from reducing the cost of production. Consumption benefits are those which result from increasing the value of a product. The description of techniques of estimating, tangible and intangible, costs and benefits will be left to other authors more proficient in the field. The cost/benefit criteria will be addressed on the basis of costs, production benefits, and consumption benefits.

It is not suggested that a rigorous economic cost/benefit analysis be performed each time a decision, with respect to an optimum approach to data updating, must be made. It is suggested that the principles applied in determining costs and benefits can be used to lend structure to the decision making process.

7.3.1 Costs

Costs are perhaps the easiest component of cost/benefit comparisons to measure. In data updating, the following costs must be addressed:

1) **Computer processing and storage resources:** As is shown in Figures 7.1 and 7.2, data updating requires processing time and disk storage. The figures also indicate that the requirements can vary significantly between the approaches to updating. An estimate of the resources needed for a given approach to satisfy the update requirement must be compared to the surplus computing resources which currently exist within the land information system. It is helpful to note that most operations for the extraction and application of propagation updates can be run as batch jobs in off-peak hours. If the surplus resources are adequate to accommodate the updating requirements then the opportunity cost of using these resources for updating versus a competing use may need to be examined. If the resources are not adequate, then system upgrades to accommodate updating must be costed. It is important to consider the future requirements of the updating system and examine these in light of the upgrade path of the computing system. Due to our limited experience in digital data updating, it may be very difficult to estimate future requirements for updates.

2) **Overhead in creating and maintaining the tracking and recording facilities for updates:** Existing data collection and updating procedures

will require alterations to accommodate the particular update approach chosen. Unless the updating system is totally transparent to the user, it will entail additional time, effort, and resultant costs. An updating system which is transparent to the user will trade-off savings in overhead costs with the cost of increasing the sophistication of the system.

3) **Software development for new update tools:** There may be an approach to updating which satisfies the update requirements more efficiently than the other approaches but the land information system does not have the required tools to implement the approach. It may be cost effective to develop these tools, especially if the savings resulting from the approach will be realized over a long period of time. The success of this development will depend upon the facilities existing in the land information system to integrate customized software. If the land information system has an in-house software development group which creates customized software for the system, they will be able to provide estimates of the cost of development. If there is no in-house software development expertise and the land information system is a commercial product, the vendor must be contacted to determine the feasibility and cost of the development. Alternatively, the development may be contracted to an outside group. It is important to consider, not only the initial development cost, but the requirement for ongoing software enhancement and, at the very least, maintenance.

4) **Manual intervention in recording, extracting, and applying updates:** A variety of approaches to updating, particularly those with limited technical sophistication, require manual intervention in the data update process. Care must be taken when estimating times for manual intervention. The experience of Huxhold et al. [1982] indicates that their best estimates for interactive graphics work should have been doubled.

5) **Data transfer of propagation updates:** As is discussed in Section 5.4, data transfer can be performed via a variety of options. Each of these options may utilize a very different costing structure. For example, transfer via magnetic media has a relatively fixed cost regardless of the file size,

transfer via a circuit switched communications line is costed according to line speed and connect time, and transfer via a packet switched communications system is costed according to the volume of data transferred. The determination of an appropriate data transfer technique must consider the timeliness requirement for the propagation updates, the volume and frequency of updates, existing facilities for data transfer, and the cost.

When considering the costs of data updating in a land information network, it is important to go beyond the current requirements and examine the load which the system will be expected to service in the future. The future requirements must then be incorporated into the planning of man power allocations and computer system upgrade paths.

7.3.2 Production Benefits

Production benefits are the simplest of the two types of benefits to measure. Unfortunately, in most instances the production benefits alone do not exceed the costs. Production benefits can occur as a result of the following improvements in the update process:

1) **Reduction of duplicate effort in maintaining data (source updating)**: A 1981 enquiry into government mapping in South Australia reported that:

> "... a single alteration to the cadastral framework could be reflected up to 35 times on different graphical products maintained by individual agencies." [SEDUNARY, 1986, pp.11]

It was also estimated that 64 man-years were expended annually on the cadastral update function. From this it can be seen that significant savings would flow from an efficient and systematic data updating service.

2) **Reduction of effort in retrieving up-to-date information:** A Western Australian Government study in 1978 estimated that:

> "... some 425 man years were required annually to service land information retrieval, again due to extensive duplication of records and effort." [SEDUNARY, 1986, pp. 4]

Again the effort involved could be significantly reduced if an updating system was in place which ensured that the required level of timeliness for updates was fulfilled.

7.3.3 Consumption Benefits

Consumption benefits are difficult to measure but are a necessary component of the cost/benefit analysis. Consumption benefits can occur as a result of the following improvements in the data update process:

1) **Availability of more timely updates:** By increasing the timeliness of the data in a land information system, the potential exists for managers, using the system, to make operational decisions on the basis of a better representation of reality than with less current information. As well, decisions which require the most up-to-date data will be sped up by decreasing the time lag in retrieving current data.

2) **Flexible provision of updates in a variety of forms:** As is detailed in Section 4.4.5, various approaches to recording source updates are capable of producing a number of types of propagation updates. For example, block source updates are capable of producing bulk, block, and possibly alert propagation updates. If a land information system is responsible for acting as a source system for a wide variety of users, e.g. a hub system, then the ability to provide propagation updates in a variety of approaches could result in significant savings for the destination systems. Thus, a destination

system could receive updates in manner which would maximize its performance criteria and/or minimize its costs in developing the capability to integrate the propagation updates into its database.

3) **Provision of data suitable for time series analysis and archival:** This benefit applies specifically to the incremental updating with attributes approach. As is mentioned in Section 5.2.2.4, incremental updating stores the incremental changes which occur to a file and corresponding time-stamping information. This automatically provides the facility to easily archive the changes which occur to a file and at a later date to re-construct a view of the database as it was at a given time. The issue of archiving changes made to digital databases is one which, like the data update problem, has not received widespread attention. Hunter [1986] gives guidelines for determining which non-current data should be archived and suggestions on appropriate structures and formats for archival.

7.4 Systematic Approach to Determining an Optimum Data Update Policy

The following section describes a suggested approach to systematically determine an optimum update policy. The process should be an iterative one where general requirements are addressed initially and as the number of suitable update approaches narrows more detailed investigations are carried out. The steps in determining an optimum data update policy are:

1) **Determine the environment in which the data update processing must function:** The environment for data update processing involves the characteristics of the updates, timeliness requirements which must be fulfilled, restrictions on the system's processing and storage resources, value of the benefits which can be achieved via improved data updating, and restrictions on the financial resources which may be committed to data updating.

2) **Determine whether capabilities exist to support the fundamental updating facilities:** The fundamental updating facilities are the abilities to: 1) track updates, 2) reformat updates, 3) transfer updates, and 4) verify updates. The requirements for these facilities are fully defined in Chapter 5. With the possible exception of verifying updates, all of these fundamental updating facilities must exist in order to perform updating via any approach. If these facilities are missing then their inclusion in the land information system must be dealt with prior to evaluating particular approaches to data updating.

3) **List the pros and cons of each of the 4 approaches to updating:** This involves listing the capabilities which may be required in the land information system to implement each approach, and listing the performance advantages and limitations of each approach for the existing data update environment.

4) **Determine the central trade-offs which must be made in the selection of an approach to updating:** This involves an examination of the pros and cons listed in step 3, the selection of the most favourable options, and the determination of the central trade-offs in choosing one approach above the other. For example, if a land information system supports block updating and not incremental updating, a decision must be made as to whether the development cost in implementing an incremental update approach would be offset by the savings in the volume of data which must be transferred in an incremental approach over a block update approach.

5) **Selection of an update policy or alteration of the update environment:** Once the critical issues in the selection of an approach to updating are highlighted, a decision must be made to select an approach or to modify the requirements of the update processing environment. For example, in some situations the relaxation of the timeliness requirement may enable the utilization of a less costly approach to data updating. Once the requirements have been changed, the whole selection process must be re-examined in light of the new requirements.

This iterative approach to the selection of an optimum update policy is necessary because of the large number of options which are available. The evaluation of options based on the general criteria given in this chapter will assist in reducing the number of options, but in the final selection the decision maker must address the specific requirements of each approach as described in Chapter 5.

7.5 Summary

In summary, the criteria for the selection of an optimum update policy are based on the capabilities of the user's system and the level of performance required of the data updating process. Trade-offs will occur between the capabilities and performance, on the basis of the costs and benefits. The user must decide which factors are to be optimized and their relative priorities.

The number and wide variability of factors involved in determining an optimum data updating policy for a land information system make it impossible to develop specific criteria to assist in the selection of an updating policy. Therefore, general criteria have been presented to assist in reducing the available options to a reasonable number which can then be investigated in detail. An understanding of the requirements of each approach to updating, as outlined in Chapter 5, is necessary to perform the required detailed investigations.

In determining which update approaches will be supported in a land information network, the system managers of the land information systems comprising the network must co-operate to develop a data update policy which is of maximum benefit to all systems across the network. This requires a willingness amongst system managers to compromise their own priorities in the interest of the wider community or it requires a co-ordinating body with the authority to impose an update policy once a consensus has been reached.

Chapter 8 CONCLUSIONS AND RECOMMENDATIONS

The conclusions which can be drawn from this research will be presented in the following three categories:

1) Conclusions: statements which can be made about data updating in a land information network based on this research.
2) Recommendations: advice for land information system managers and land information network co-ordinators on addressing the issue of data update.
3) Future Research: areas worthy of further research which naturally arise from questions and difficulties which were encountered but not addressed due to limited time and resources.

8.1 Conclusions

1) The model of data updating in a land information network developed in this research has been validated by the results of the prototyping. The data update process, as presented in Section 4.3, has proven to be a viable procedure for the propagation of data updates.
2) The update tracking database can be readily implemented as an extension of a land information network's data directory.
3) The approaches of bulk, block, incremental, and alert updating are capable of providing a variety of solutions to the data update problem; covering a wide range of capability and performance criteria. The techniques developed may be implemented by manual or computerized means.
4) The bulk update approach has been shown to be a simple and effective form of updating with trade-offs in the high volume of data which must be transferred. It is a required facility in any land information network as it provides the means for the initial transfer of data sets.

5) The block update approach has been shown to provide an efficient means of propagating updates which occur primarily in localized areas. It may be readily implemented on systems which currently associate an attribute with each feature to identify the district or area it is a part of. It is important to closely examine the characteristics of the updates before choosing a block update approach.

6) The incremental update approach has been shown to be very efficient in the volume of data to be transferred at the expense of processing resources. It is ideal for updates which are low to moderate in number and widely spatially distributed. Recording incremental updates by attaching time-stamping attributes to each feature will also enable the creation of a view of the database as it appeared at any time in the past. This provides the facility for time series analysis.

7) The alert update approach has been shown to be a simple, highly compact, and economical means of highlighting areas where updating has occurred. It is well suited for situations where a high degree of timeliness is required and a bulk update facility exists.

8) The verification of the correct application of propagation updates can be efficiently and rigorously performed by determining if the updated data is topologically clean. This does not require the transfer of the topological structure associated with the data; only that source data be topologically structured and clean. Thus, it is advantageous to, where possible, store data in a topologically clean form.

9) Due to the variability of the data updating environment, it is not possible to produce specific criteria for the selection of optimum update policies. This thesis presents general criteria which can be used, in an iterative process, to reduce the number of available options and thus enable detailed investigations of the few most suitable approaches.

8.2 Recommendations

1) Designers of land information systems must provide facilities to record when updates occur on data sets and to track the origin, currency, and quality of all data sets within the system, if data is to be confidently exchanged and updated.

2) All users of land information systems who are involved in, or pursuing, the interchange of data should include discussions of the subsequent updating issue in their negotiations for the data.

3) Land information system managers need to examine the alternatives to bulk updating, namely block, incremental, and alerts, and determine whether their systems could readily support or utilize these approaches to the benefit of themselves or their users.

4) The land information community should seriously consider its commitment to the concept of a land information network and take positive steps to address the data updating issue. If the current laissez-faire attitude continues, the community runs the risk of data sharing becoming merely a one-off transfer with the continuance of duplication of effort in the maintenance of its data sets.

8.3 Future Research

1) Further information about the various approaches to data updating could be obtained by testing the prototype system with a very large data set containing convoluted line work and with a non-topologically structured data set.

2) Insight into the impact of the capability criteria would be gained from an investigation of the extent to which the required data update tools exist in commercial land/geographic information system software products.

3) Research into the data update environment of various applications of land/geographic information systems could yield criteria for the selection of

optimum data update policies for particular applications. This would require an examination of the number, volume, spatial distribution, and timeliness requirements for data updates within specific application areas.

4) The ultimate test of the generality of the functional description of the data update model would be its implementation in an operational land information network utilizing two, or more, heterogeneous land information systems.

Chapter 9 REFERENCES

ALLDER, W.R. and A.A. ELASSAL [1983] "U.S.G.S. Digital Cartographic Data Standards. Digital Line Graphs from 1:24,000-Scale Maps", United States Geological Survey circular 895-C, Alexandria, Virginia, U.S.A.

ARCHER, D.W. [1986] "Integrating Non-Graphic Databases", Unpublished paper presented at the AURISA New South Wales Seminar, Australian Urban and Regional Information Systems Association, Sydney, Australia.

BEDARD, Y. [1986], "A Study of the Nature of Data Using a Communication-Based Conceptual Framework of Land Information Systems", Proceedings of International Federation of Surveyors XVIII Congress, Commission 3, Toronto, Canada.

BELL, K.M. [1987] "State Land Systems Development in Queensland", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

BLACHUT, T.J.; A. CHRZANOWSKI and J.H. SAASTAMOINEN [1979] Urban Surveying and Mapping, SPRINGER-VERLAG, New York, U.S.A.

COMMONWEALTH LAND INFORMATION SUPPORT GROUP (AUSTRALIA) [1987] "Land-Related Data Directories", LINC (Land INformation Chronicle), Vol. 1 Number 2, Canberra, Australia.

COVER, C.; D. HUTCHISON and E. SIEKIERSKA [1983] "Data Base Management - Discussion Paper", Internal Report, Surveys and Mapping Branch, Energy, Mines and Resources, Ottawa, Canada.

DAW, W.E. [1986] "Benefits of Land Information Systems", Australian Advisory Committee on Land Information, Technical Paper 86/06, Canberra, Australia.

DEPARTMENT OF LANDS (NEW SOUTH WALES) [1987] Tender No. 1262 For the supply, installation, setting to work & maintenance of: Hardware and software for the first stages of the New South Wales land information system (LIS) comprising sub system (A) data capture, maintenance and update of the graphic data base and sub system (B), LIS hub spatial referencing system. Sydney, Australia.

DEPARTMENT OF LANDS (SOUTH AUSTRALIA) [1986] "Digital Cadastral Data Base: Data Collection Phase", Report by the DCDB Project Team, Library Reference 18/86, Adelaide, Australia.

EDDINGTON, R.A. [1987] "Landata Status Report", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

ENVIRONMENTAL SYSTEMS RESEARCH INSTITUTE [1985] ARC/INFO Users Manual, Redlands, California, U.S.A.

FEELEY, J. [1986] "The Importance of Telecommunications and Standards in handling Digital Map and Chart Information", Proceedings of CPA/CHA Colloquium IV: Land, Sea and Space - Today's Survey Challenge, Canadian Institute of Surveying and Mapping, Lake Louise, Alberta, Canada.

GAUDET, R.J.; C. CARLIN and J. MCLAUGHLIN [1984] "Property Mapping in Support of a Parcel-Based Land Information Network", Proceedings of Urban and Regional Information Systems Association: The Changing Role of Computers in Public Agencies, Seattle, Washington, U.S.A.

GOH, P.C. [1987] "An Entity-Relationship Spatial Model For Land Information Systems", M.Surv. Thesis, School of Surveying, University of New South Wales, Sydney, Australia.

GRSKOVICH, E.N. [1978] "Continuous and Instantaneous Updating of a Land Data System", Proceedings of the Second MOLDS Conference: Land Data Systems Now, Washington, D.C., U.S.A.

HART, A.L. [1987] "New South Wales Land Information System: Status Report 1987", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

HENCO [1985] INFO VAX User's Guide, Henco Software Inc., Massachusetts, U.S.A.

HEBBLETHWAITE, D. and R. EDEN [1986] "The Corporate Working Map", Proceedings of the URPIS 14 Conference, Melbourne, Australia.

HUMPHRIES, B. [1984] "The Crisis in Land Management", Proceedings from the FIG International Symposium: The Decision Maker and Land Information Systems, Edmonton, Alberta, Canada.

HUNTER, G.J. [1986] "A Guide to Procedures for Selecting Non-current Data, Worthy of Archiving, in a Land Information System", Report, Department of Surveying, University of Melbourne, Melbourne, Australia.

HUXHOLD, W.E.; R.K. ALLEN and R.A. GSCHWIND [1982] "An Evaluation of the City of Milwaukee Automated Geographic Information and Cartographic System in Retrospect", Paper from the Policy Development Information System group, City of Milwaukee, U.S.A.

HYDE, I. [1987] "Western Australian Land Information System Status Report", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

INTER-DEPARTMENTAL COMMITTEE ON LAND INFORMATION (NEW SOUTH WALES) [1983] Internal report, Sydney, Australia.

INTERNATIONAL FEDERATION OF SURVEYORS [1981] "Resolutions of the Commission 3 of the International Federation of Surveyors", Montreux.

KENNEDY, E.A. [1986] "Advanced Technologies for Development of the Alberta LRIS Network", Proceedings of CPA/CHA Colloquium IV: Land, Sea and Space - Today's Survey Challenge, Canadian Institute of Surveying and Mapping, Lake Louise, Alberta, Canada.

KIRBY, M.D. [1982] "Computers: Who is Concerned?", Proceedings of URPIS 10, Sydney, Australia.

MACIVER, M.E. [1984] "Data Sharing Methods for Land Records Systems", Proceedings of the Urban and Regional Information Systems Association, Seattle, Washington, U.S.A.

MARTIN, J. [1972] Systems Analysis for Data Transmission, Prentice-Hall Inc., Englewood Cliffs, New Jersey, U.S.A.

MOELLERING, H. (ed.) [1985] Report #6, Digital Cartographic Standards: An Interim Proposed Standard, National Committee for Digital Cartographic Data Standards, Ohio, U.S.A.

MOELLERING, H. (ed.) [1987] Report #8, Digital Cartographic Standards: A Draft Proposed Standard for Digital Cartographic Data, National Committee for Digital Cartographic Data Standards, Ohio, U.S.A.

MOREHOUSE, S. [1987] "ARC/INFO: A Geo-Processing Tool Box", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

PEARSON, M. and P. GAREAU [1986] "Exchanging Digital Base Mapping Data", Proceedings of the Second International Symposium on Spatial Data Handling, Seattle, Washington, U.S.A.

PEUQUET, D.J. [1984] "A Conceptual Framework and Comparison of Spatial Data Models", Cartographica, Vol. 21 No. 4.

PHILLIPS, M. and K.C. BELL [1987] "ACT LIS Status Report - 1987", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

PRICE, D. [1987] "Tradition of Success", Bits & Bytes Magazine, May 1987, New Zealand.

PULLAR, D.N. [1985] "The Integration of Cadastral and Natural Resource Information in the New Zealand Land Information System", M.Surv. Thesis, University of Otago, Dunedin, New Zealand.

ROBERTS, G. [1987] "Land Information Systems Developments in Tasmania", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

SEDUNARY, M.E. [1984] "LOTS, and the Nodal Approach to a Total Land Information System", Proceedings from FIG International Symposium: The Decision Maker and Land Information Systems, Edmonton, Alberta, Canada, 1984.

SEDUNARY, M.E. [1986] "A Development Policy for Land Information Processing in South Australia", Internal report, Land Information Unit, South Australia Department of Lands, Adelaide, South Australia.

SEDUNARY, M.E. [1987] "LIS Development in South Australia: 1987 Status Report", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

SOUTH AUSTRALIAN DATA PROCESSING BOARD [1985] Minute dated 16-Oct-1985, DPB 28/80.

STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES) [1987a] The Conceptual and Technological Framework for the New South Wales Land Information System, Vol.1 "Summary", Sydney, Australia.

STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES) [1987b] The Conceptual and Technological Framework for the New South Wales Land Information System, Vol.2 "The Framework for the Land Information System", Sydney, Australia.

STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES) [1987c] The Conceptual and Technological Framework for the New South Wales Land Information System, Vol.3 "The Requirements for Land Information", Sydney, Australia.

STATE LAND INFORMATION COUNCIL (NEW SOUTH WALES) [1987d] The Conceptual and Technological Framework for the New South Wales Land Information System, Vol.4 "Technology Relevant to the Land Information System", Sydney, Australia.

STEPHENS, V. [1987] "Land Information Systems Status Report: Northern Territory", Proceedings of URPIS 15: Information for Policy Makers, Hobart, Tasmania, Australia.

TANENBAUM, A.S. [1981a] Computer Networks, Prentice-Hall Inc., Englewood Cliffs, New Jersey, U.S.A.

TANENBAUM, A.S. [1981b] "Network Protocols", Computing Surveys, Vol.13, No.4, pp. 453-489.

UNITED NATIONS ECONOMIC AND SOCIAL COUNCIL [1973] "Report of the Ad-Hoc Group of Experts on Cadastral Surveying and Mapping", Tokyo, Japan.

WEIR, C.H. and R.B. SWETNAM [1986] "Assessing the Economic Benefits to Users of L.I.S. Projects", Stewart, Weir & Co., Edmonton, Alberta, Canada.

WILSON, T. [1987] (Director, GENASYS II Pty Ltd), Personal communication.

Chapter 10 APPENDIX A: PLOTS OF TEST CASES

The following plots depict the test data used in the testing of the data update prototype, as described in Section 6.2.2. The plots of the bulk approach essentially show a copy of the updated source file. The plots of the block approach indicate, by shading, the blocks which are affected by the updates. The plots of the incremental approach indicate, via dotted lines, the features to be added and deleted. The plots of the alert updates show the alerts as point symbols.

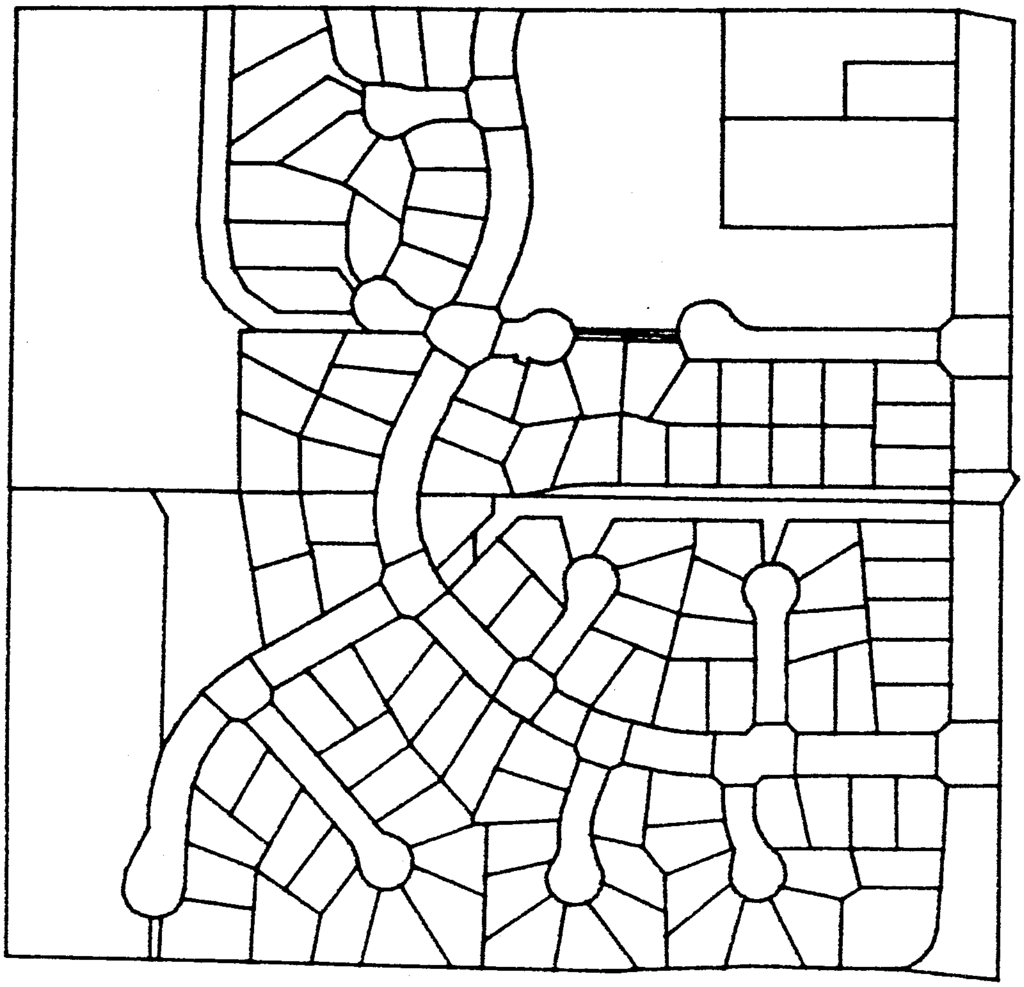


Figure A.1 **Test Area, File S60**

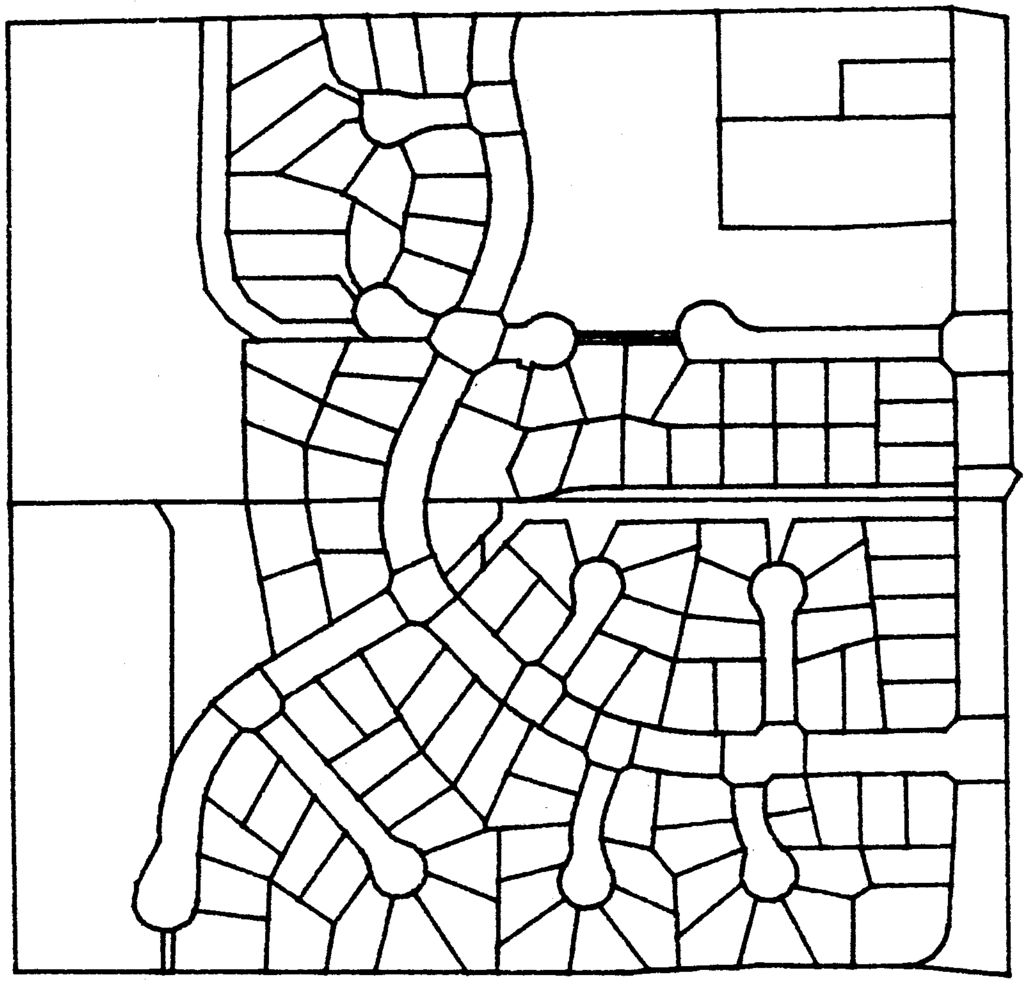


Figure A.2 **S72 Bulk Updates**

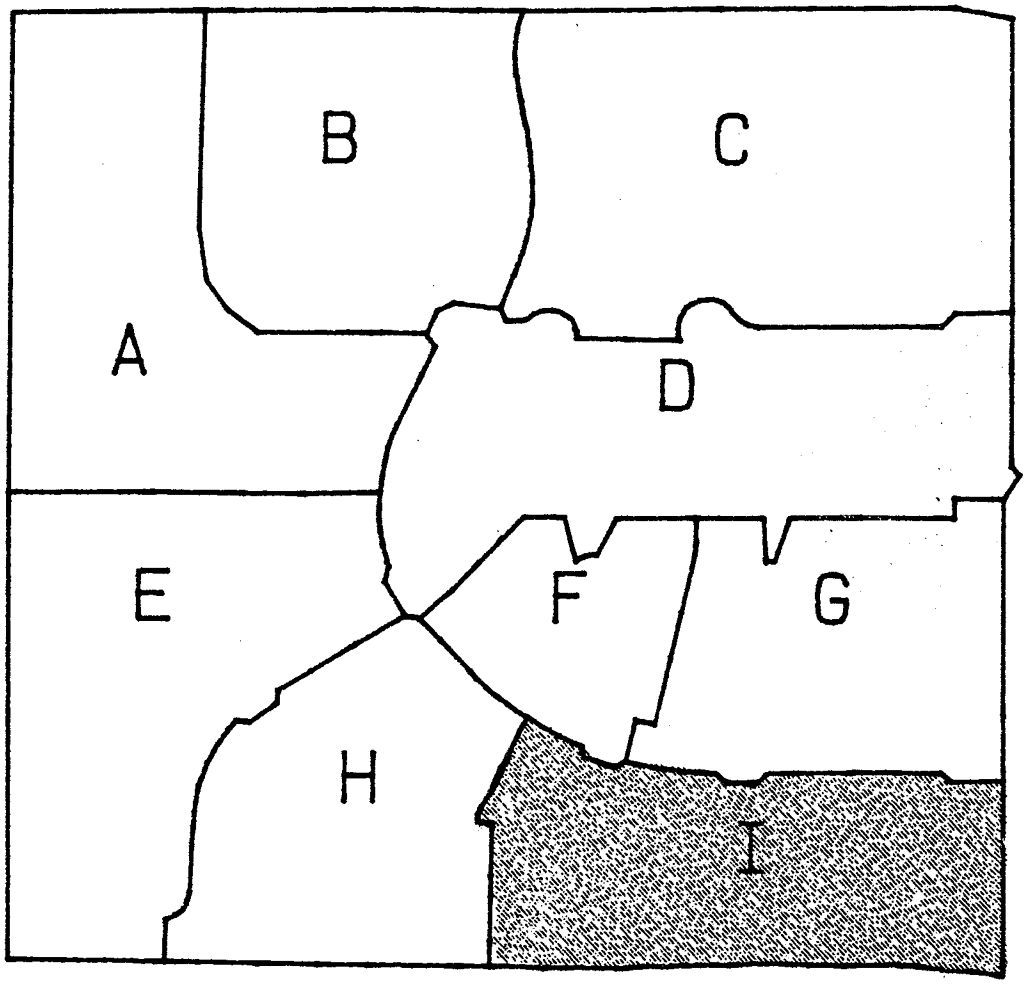


Figure A.3 **S72 Block Updates**

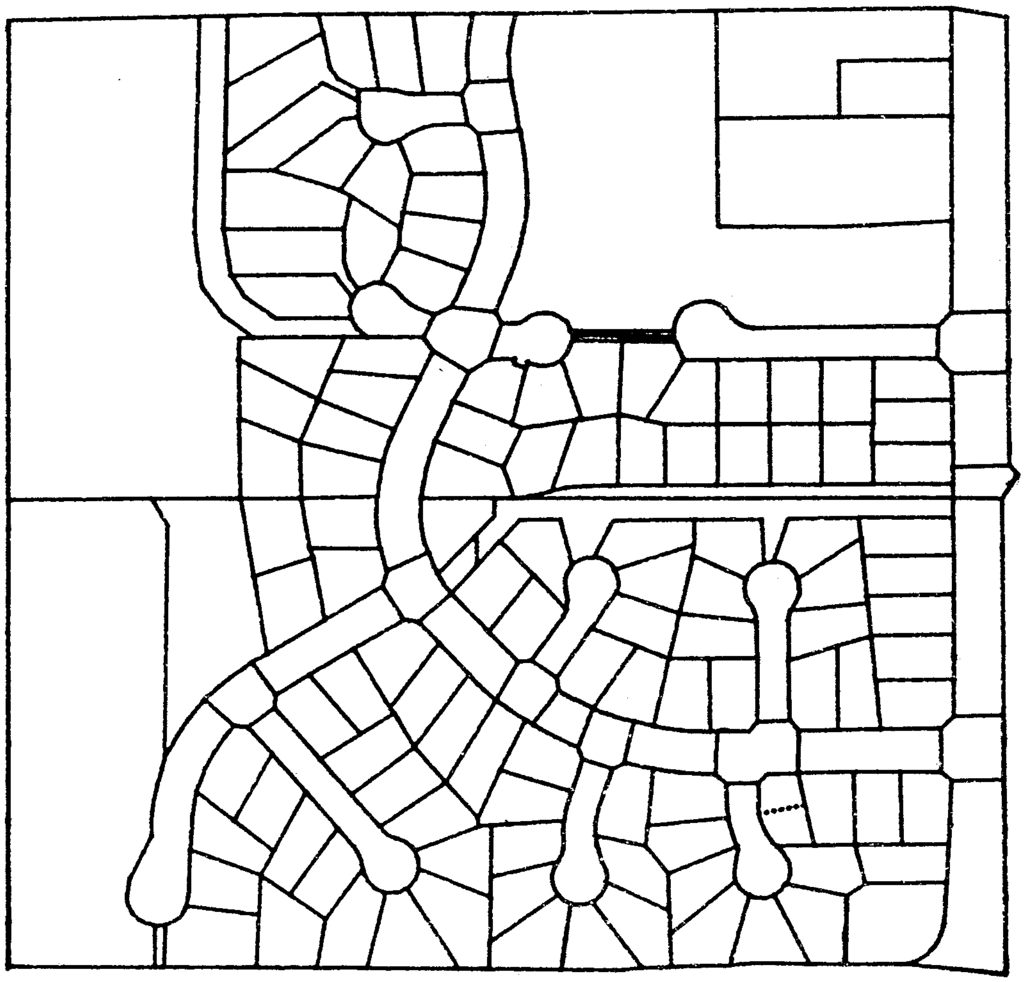


Figure A.4 **S72 Incremental Updates**

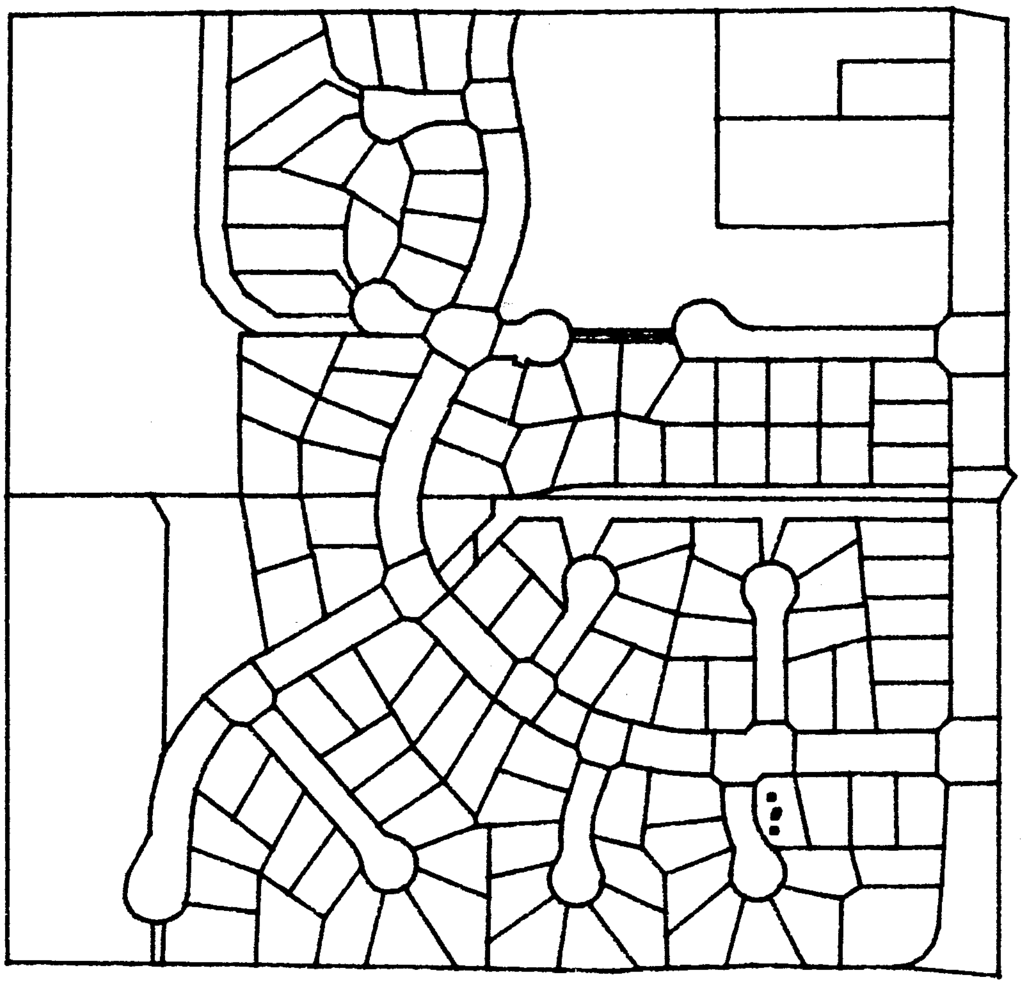


Figure A.5 **S72 Alert Updates**

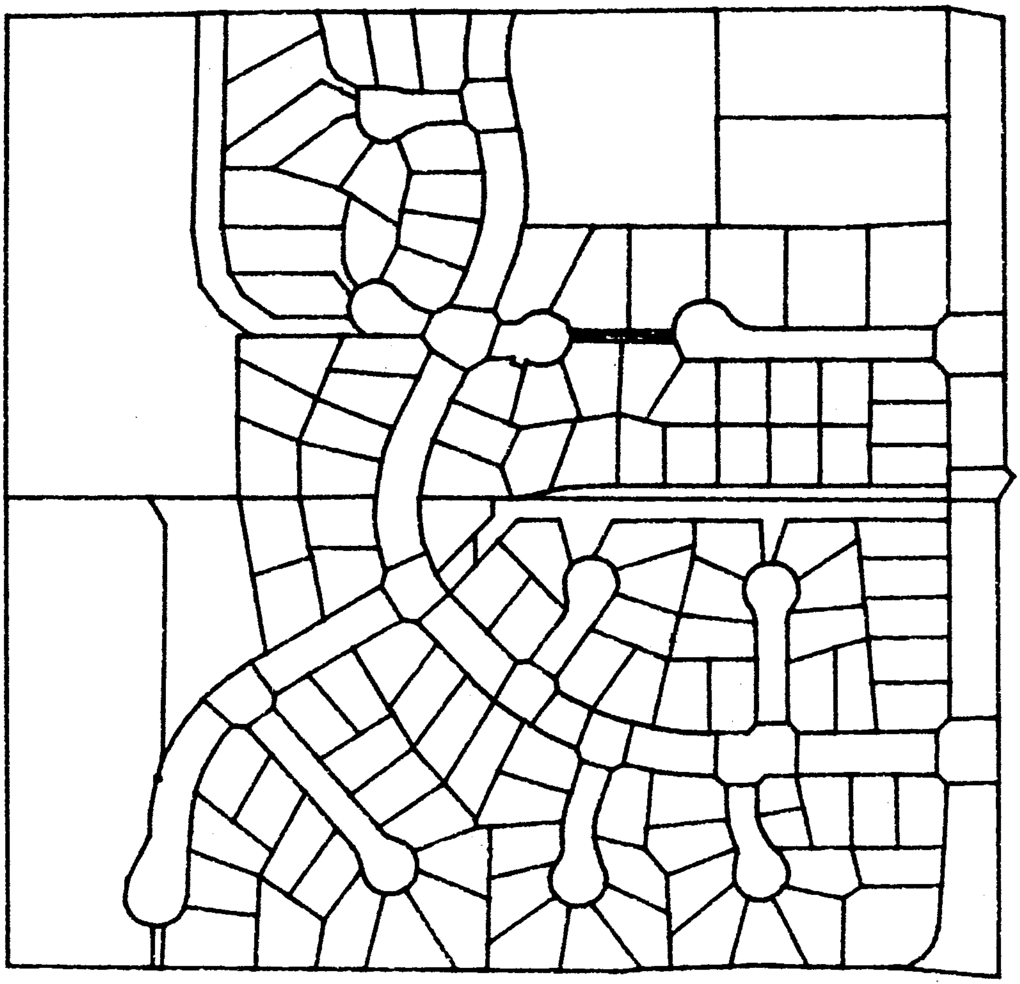


Figure A.6 **S73 Bulk Updates**

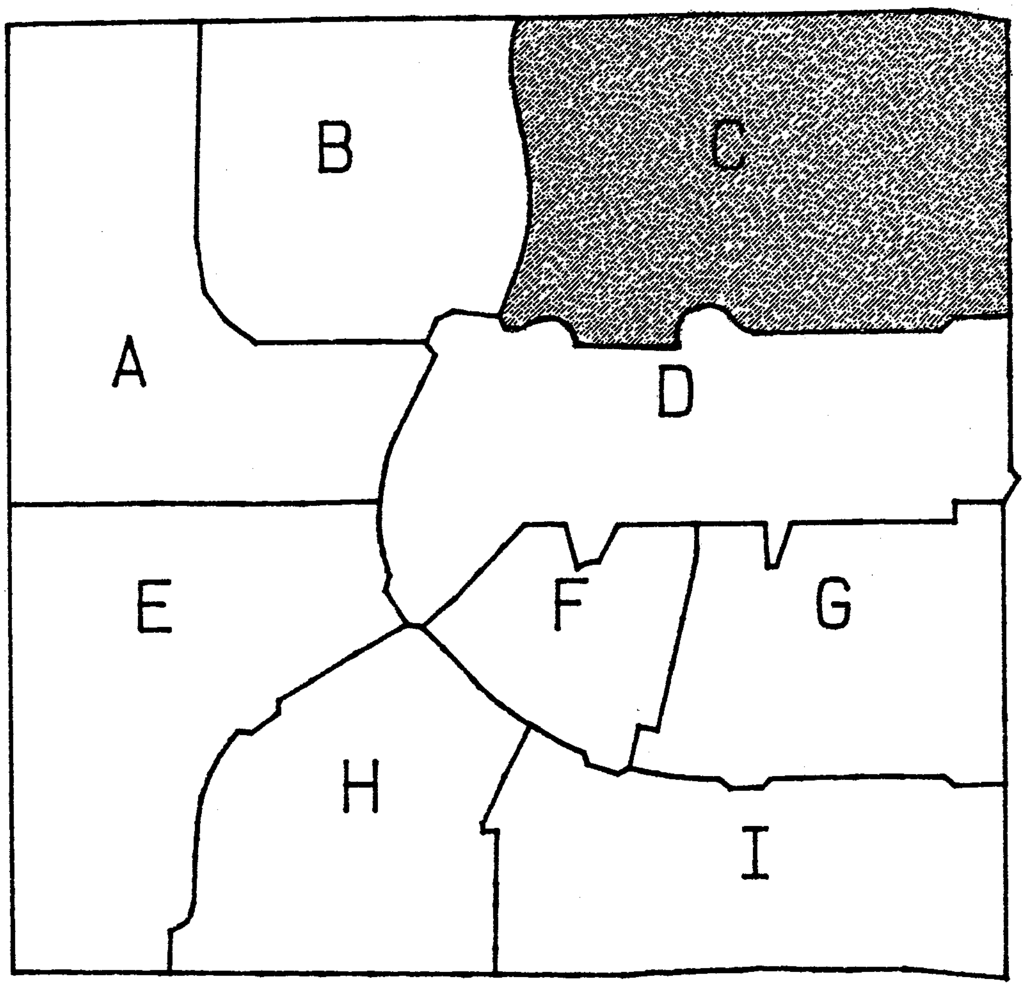


Figure A.7 **S73 Block Updates**

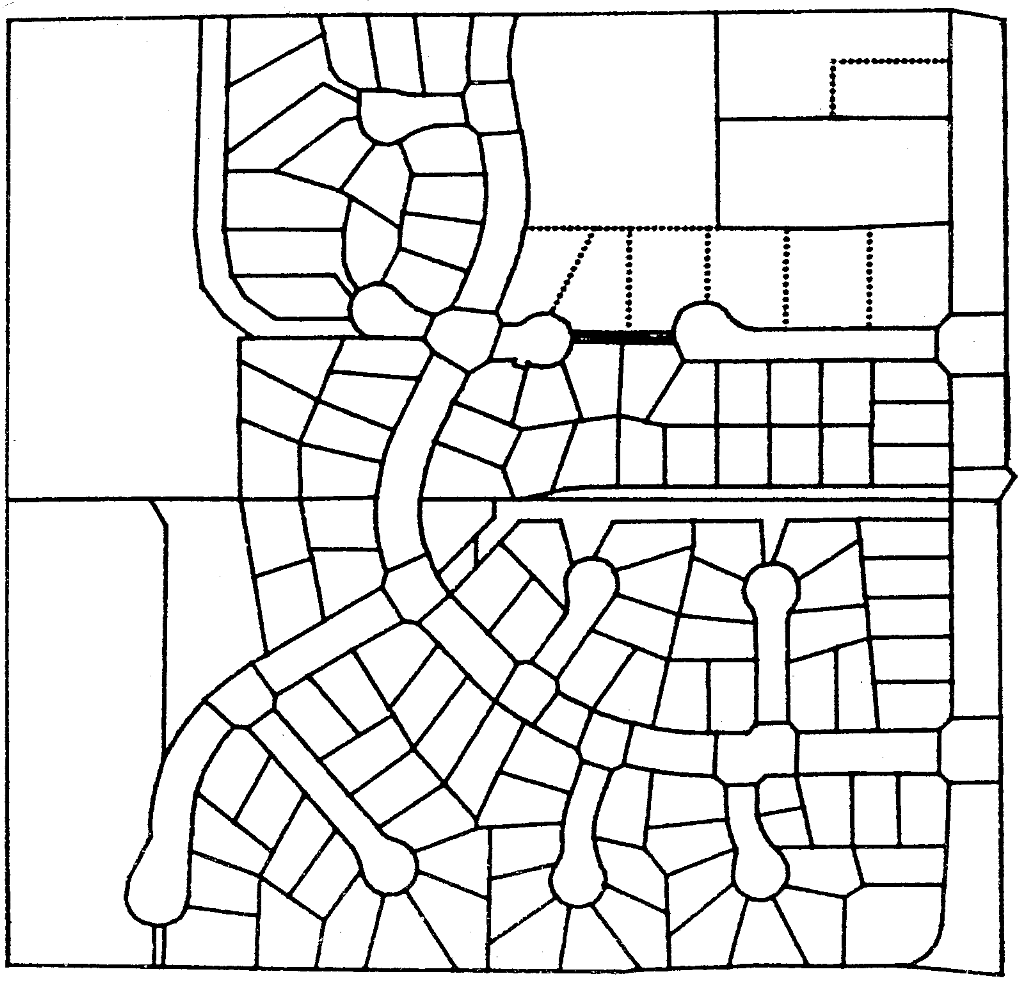


Figure A.8 **S73 Incremental Updates**

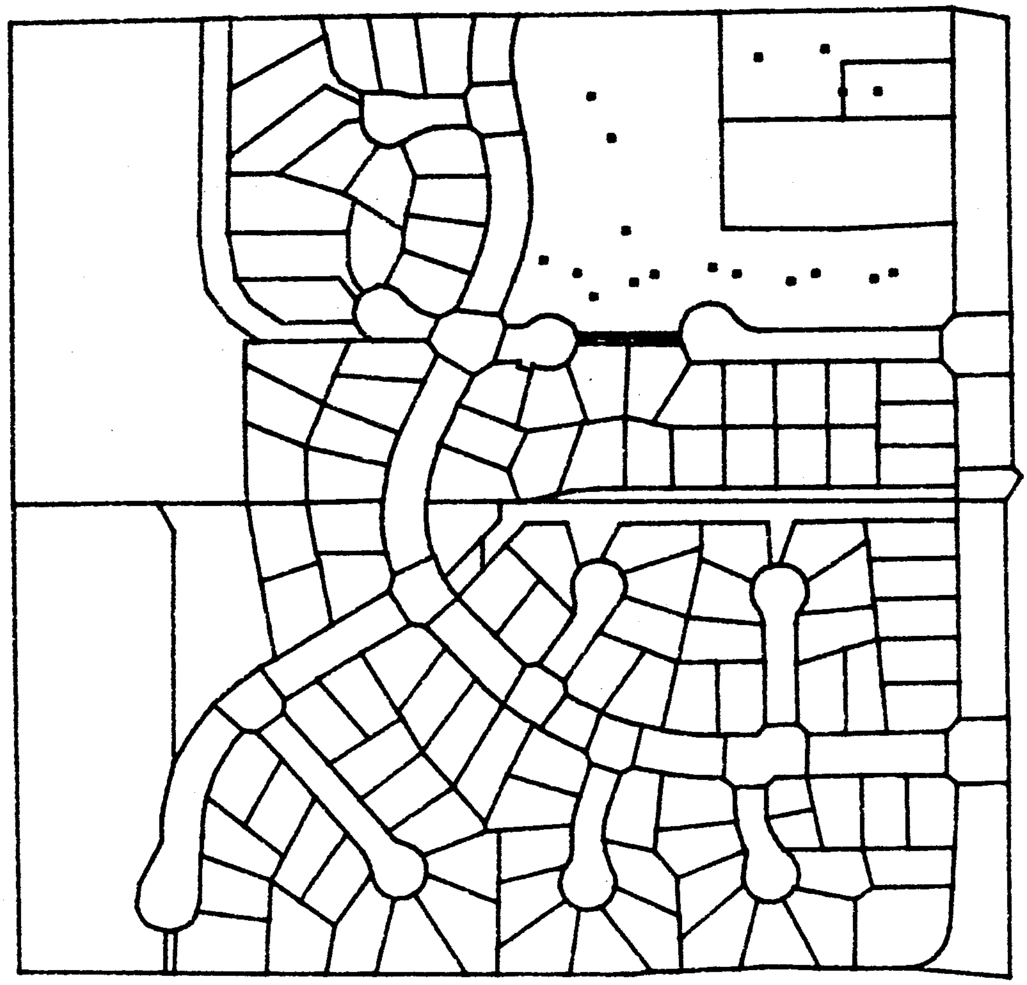


Figure A.9 **S73 Alert Updates**

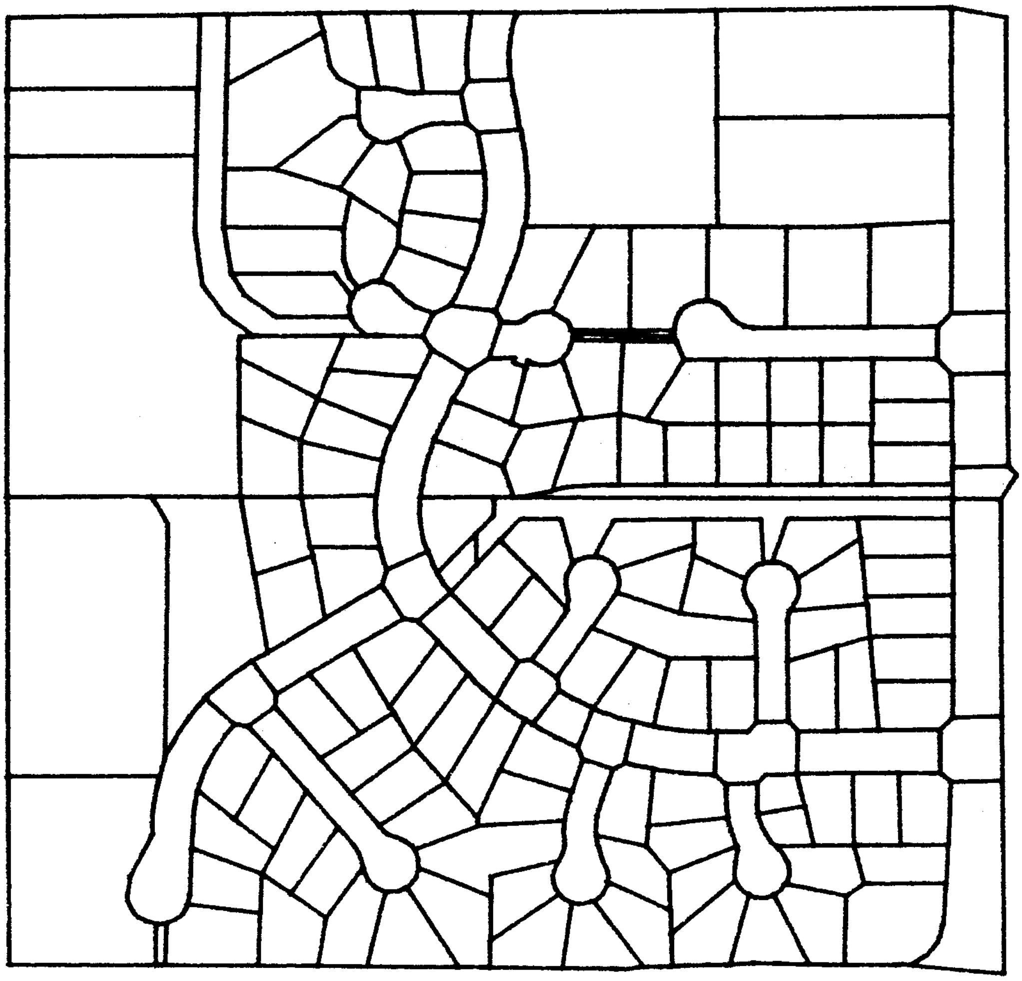


Figure A.10 **S74 Bulk Updates**

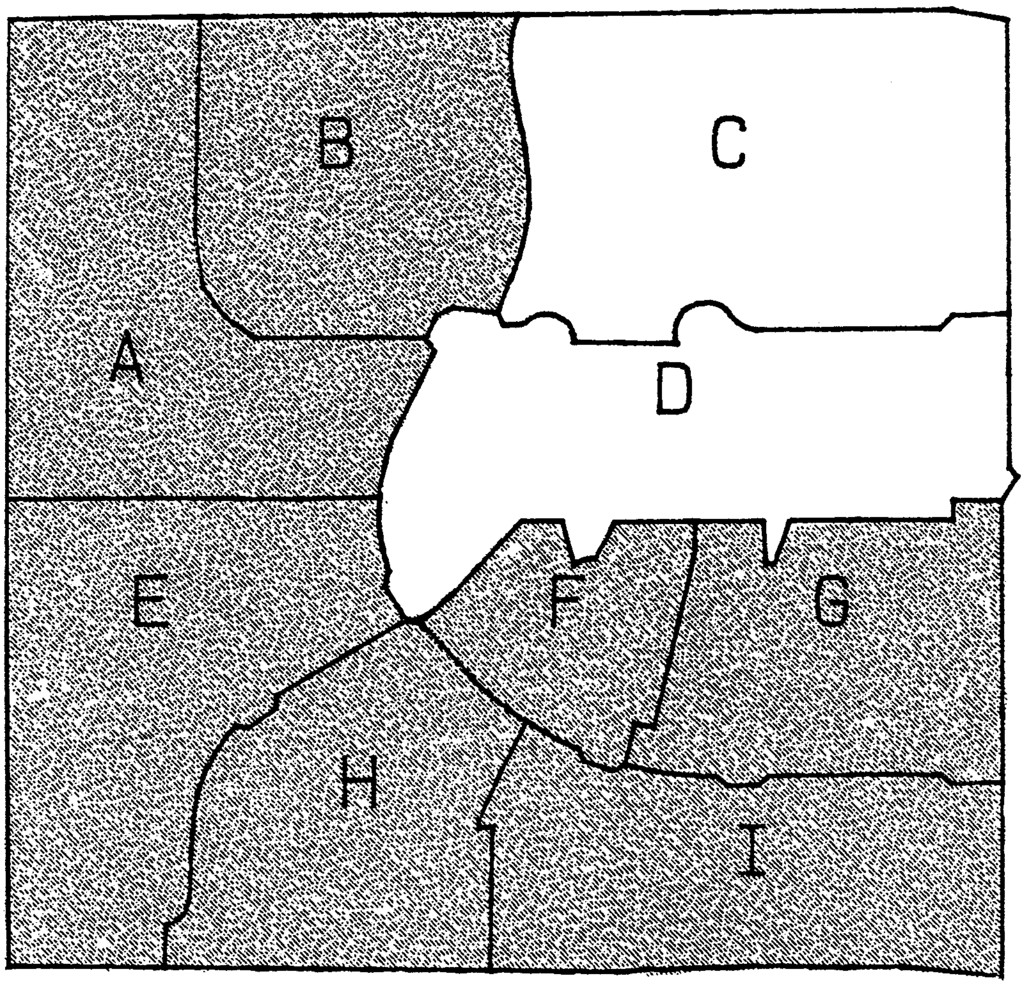


Figure A.11 **S74 Block Updates**

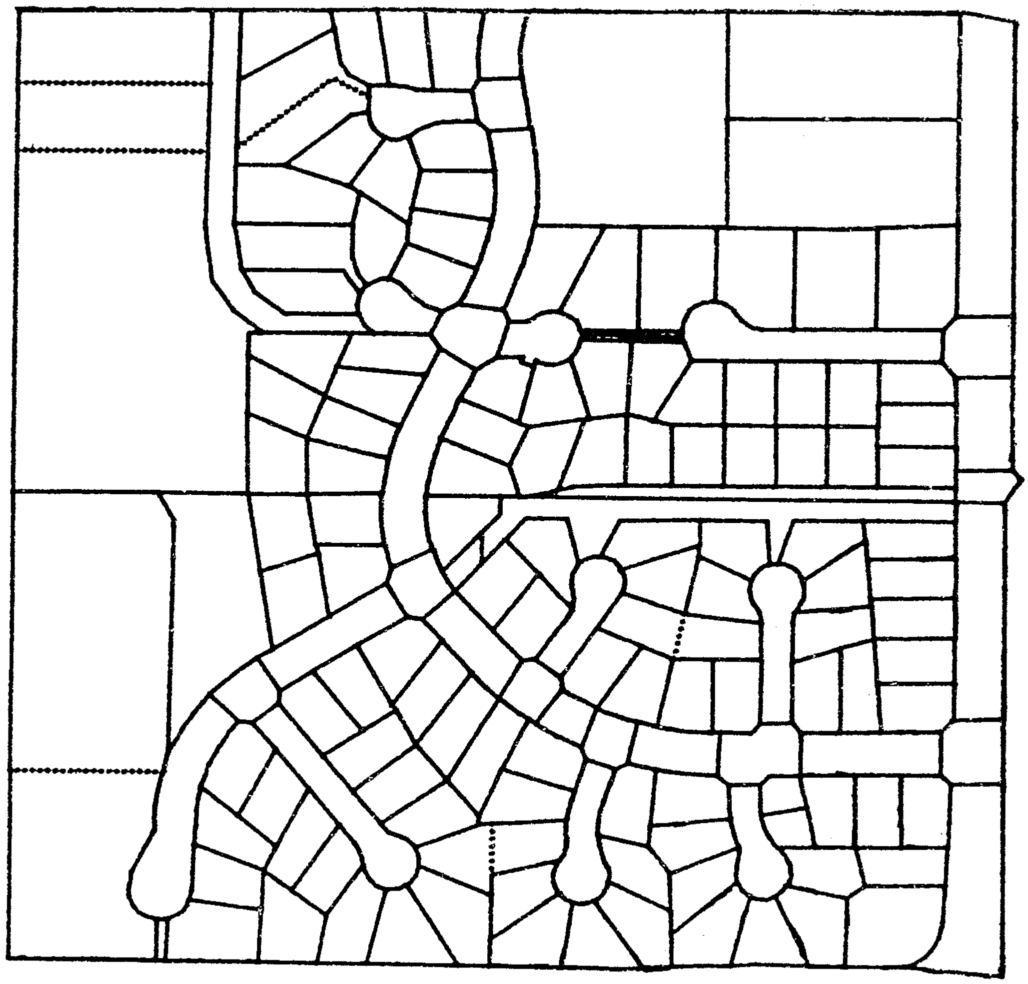


Figure A.12 **S74 Incremental Updates**

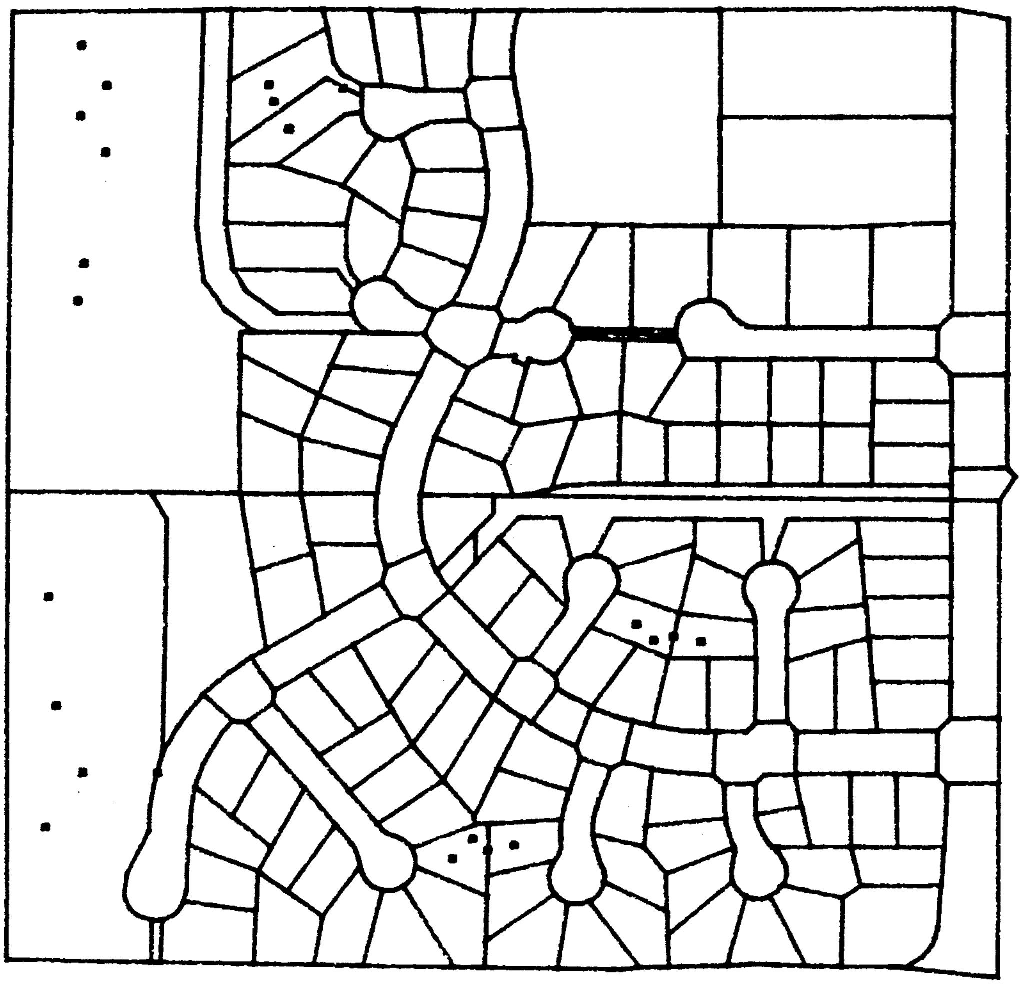


Figure A.13 **S74 Alert Updates**

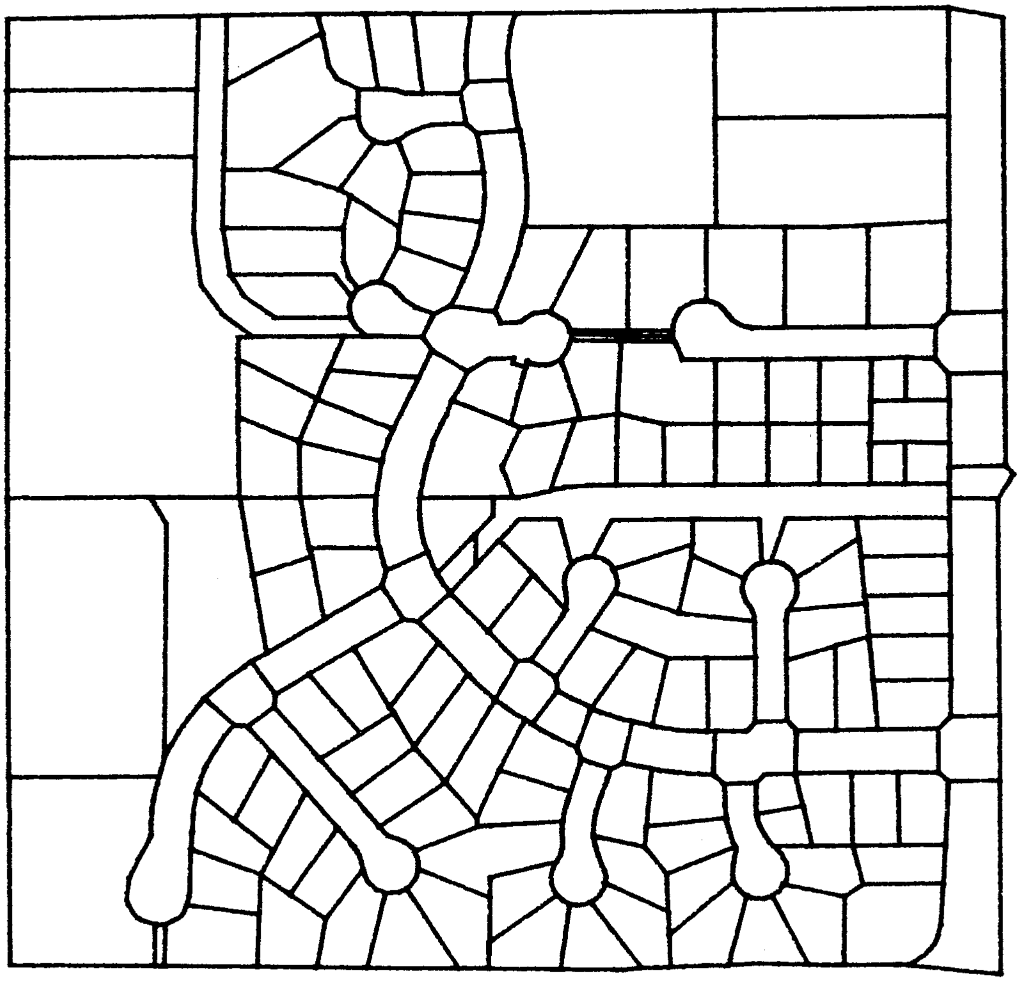


Figure A.14 **S75 Bulk Updates**

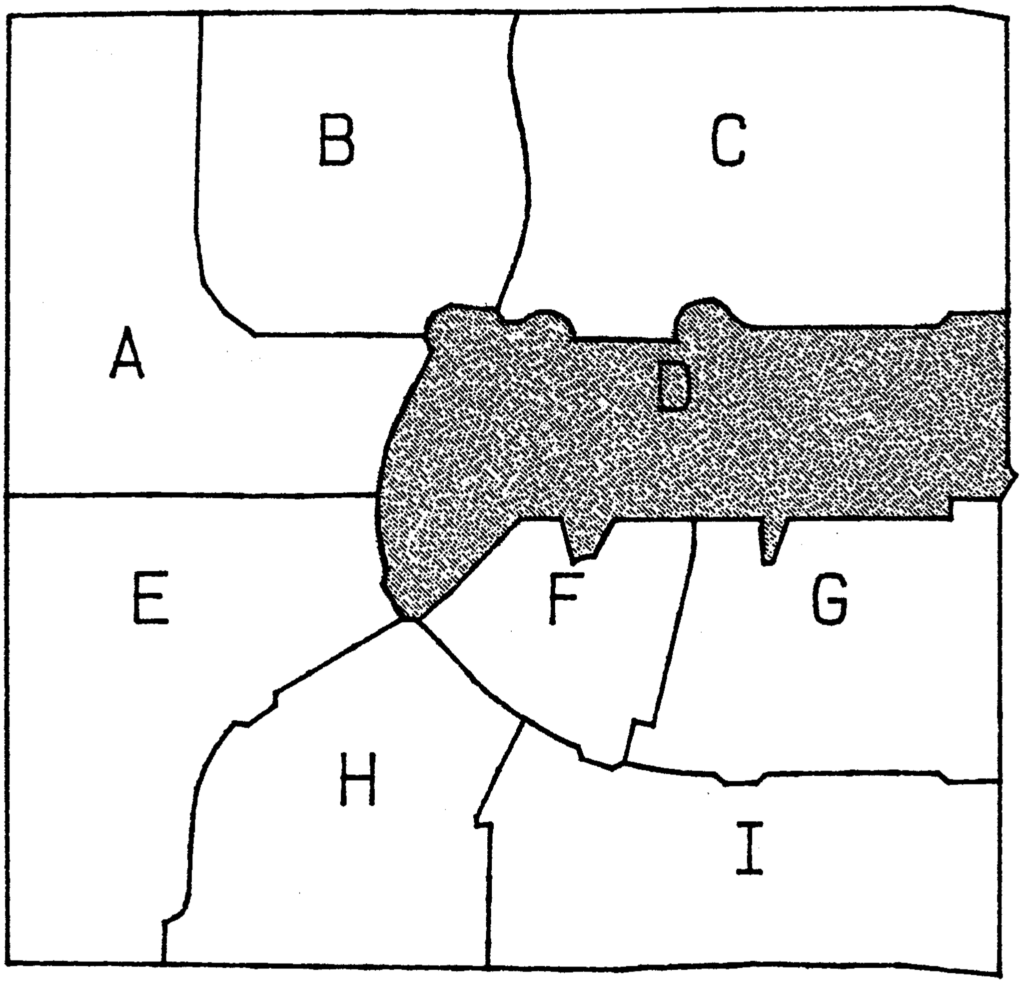


Figure A.15 **S75 Block Updates**

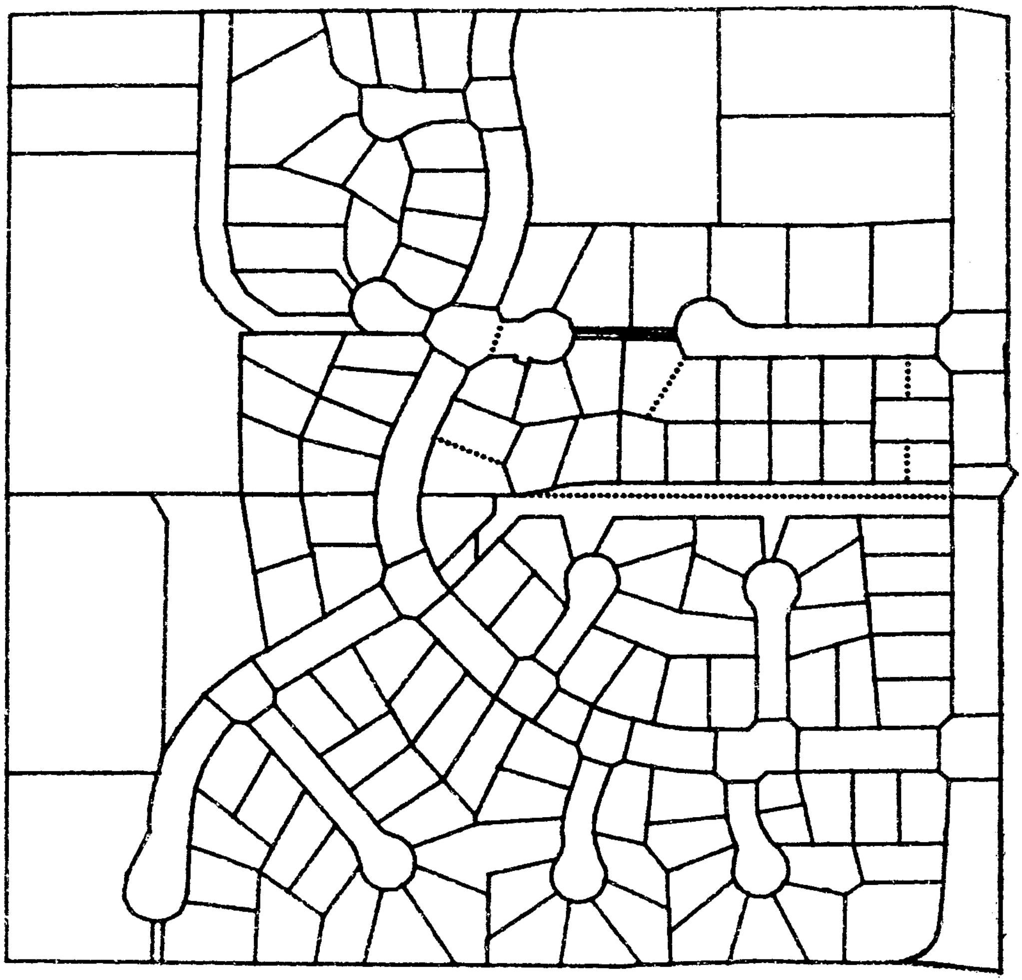


Figure A.16 **S75 Incremental Updates**

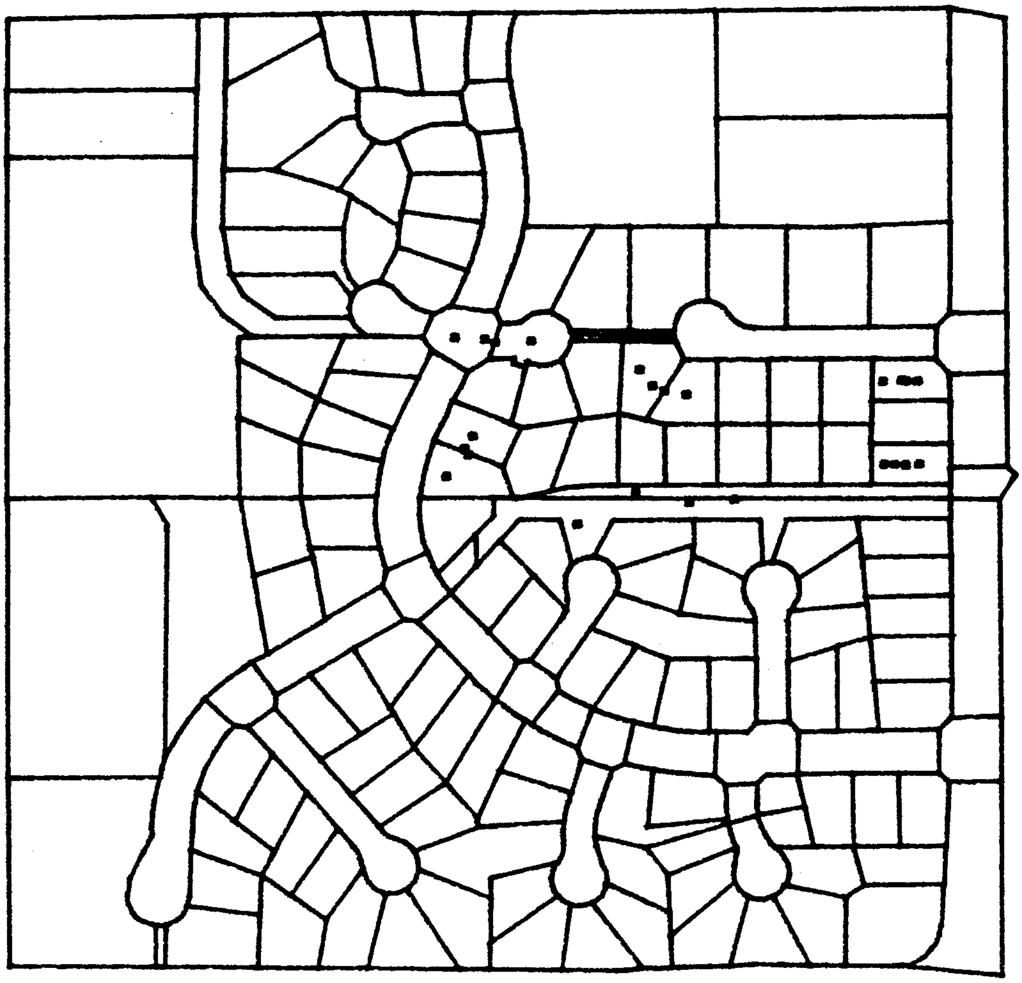


Figure A.17 **S75 Alert Updates**

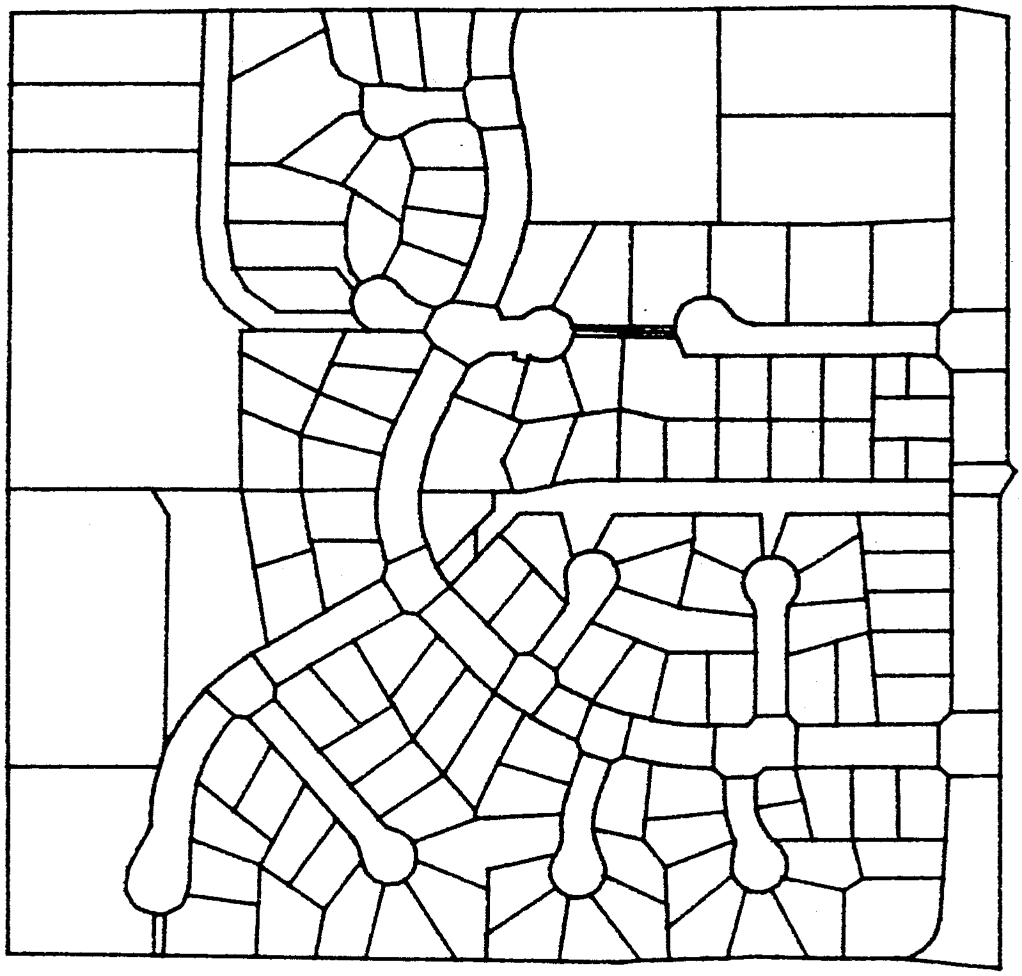


Figure A.18 **S76 Bulk Updates**

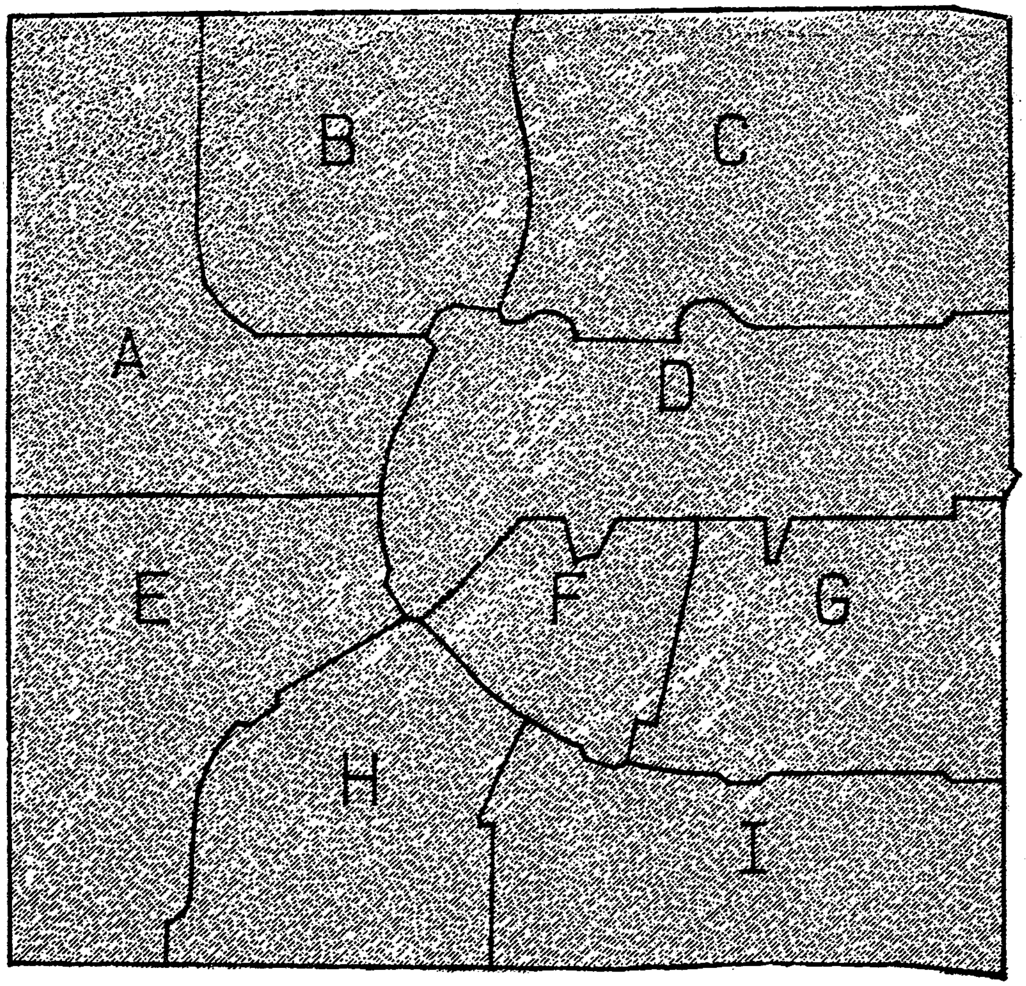


Figure A.19 **S76 Block Updates**

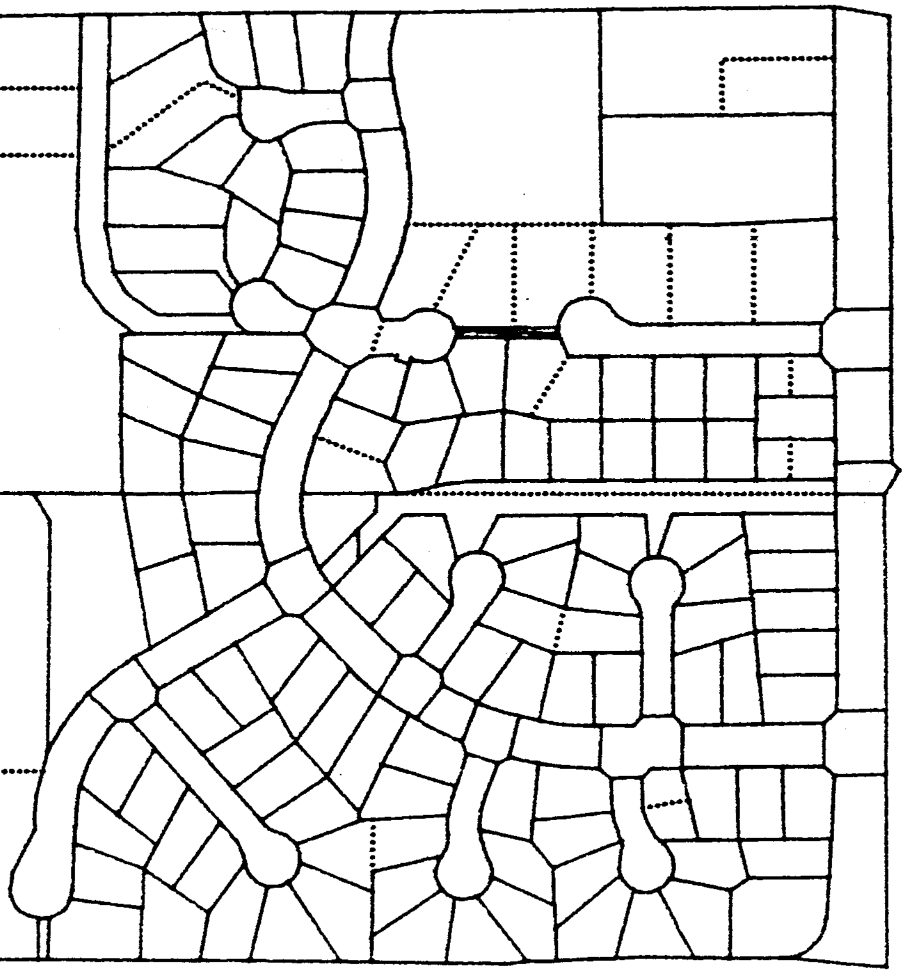


Figure A.20 **S76 Incremental Updates**

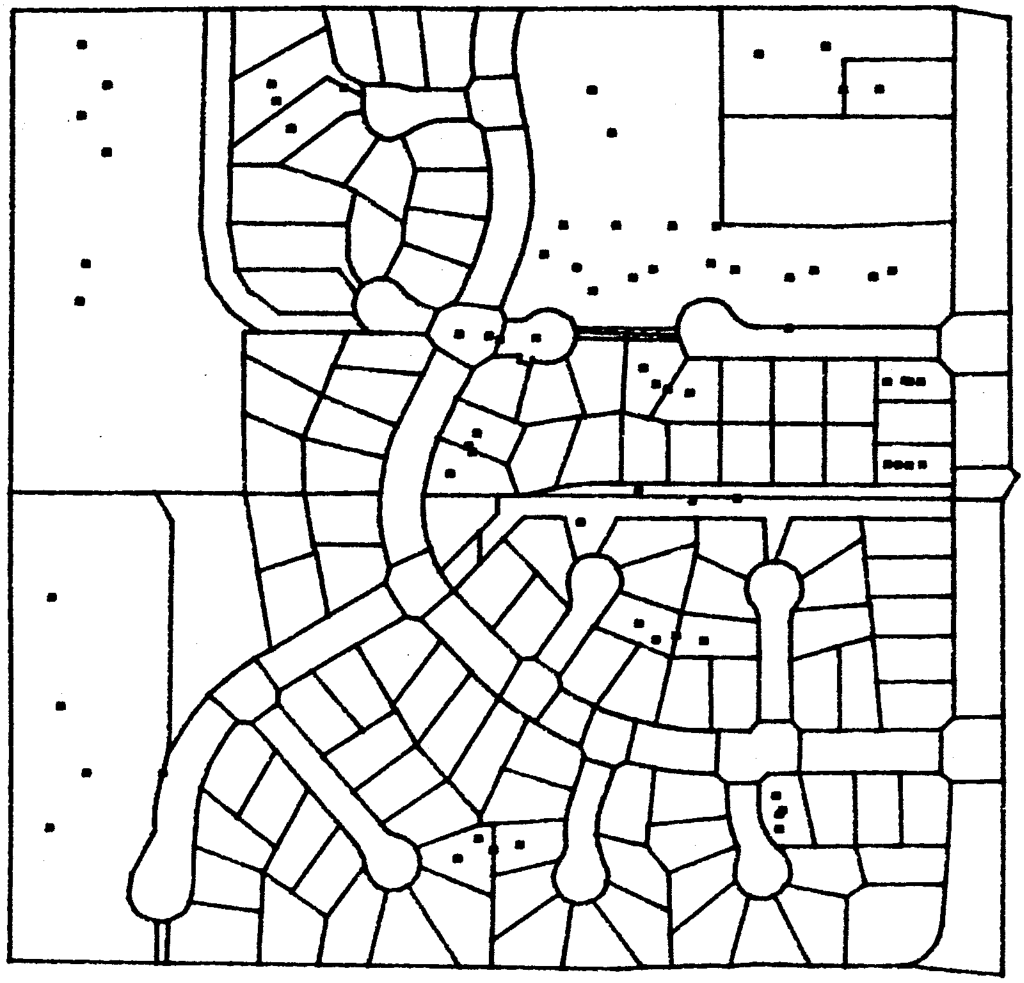


Figure A.21 **S76 Alert Updates**

Chapter 11 APPENDIX B: TABLE OF RESULTS FOR ALL TESTS

The following table shows the results of the tests of all of the approaches to data updating. This is a combination of Tables 6.12 to 6.15.

UPDATE FUNCTION	S72TIME	S72SIZE	S73TIME	S73SIZE	S74TIME
BULK EXTRACT	23	216	20	274	20
BULK EXPORT	37	216	46	274	45
BULK IMPORT	42	216	53	274	54
BULK APPLY	33	216	35	274	35
BULK CLEAN	106	216	107	274	104
TOTAL	241	216	261	274	258
BLOCK BLKINT	529	202	522	209	536
BLOCK NUMBLK		1		1	
BLOCK BLKBUL	16	1	15	1	16
BLOCK EXTRACT	206	34	204	19	216
BLOCK EXPORT	17	34	14	19	33
BLOCK IMPORT	26	34	23	19	42
BLOCK APPLY	269	34	210	19	232
BLOCK CLEAN	104	265	106	274	101
TOTAL	638	34	572	19	640
INCWOA EXTRACT	823	10	883	13	928
INCWOA EXPORT	25	10	26	13	26
INCWOA IMPORT	34	10	35	13	35
INCWOA APPLY	172	10	185	13	181
INCWOA CLEAN	105	265	103	274	102
TOTAL	1159	10	1232	13	1272
INCWA EXTRACT	426	10	376	13	423
INCWA EXPORT	25	10	26	13	26
INCWA IMPORT	34	10	35	13	35
INCWA APPLY	172	10	185	13	181
INCWA CLEAN	105	265	103	274	102
TOTAL	762	10	725	13	767
ALERT GENERATE	90	3	91	5	90
ALERT EXPORT	12	3	12	5	12
ALERT IMPORT	21	3	21	5	21
TOTAL	123	3	124	5	123

UPDATE FUNCTION	S74SIZE	S75TIME	S75SIZE	S76TIME	S76SIZE
BULK EXTRACT	280	20	280	20	280
BULK EXPORT	280	44	280	44	280
BULK IMPORT	280	53	280	53	280
BULK APPLY	280	33	280	33	280
BULK CLEAN	280	103	280	103	280
TOTAL	280	253	280	253	280
BLOCK BLKINT	211	536	210	536	211
BLOCK NUMBLK	7		1		9
BLOCK BLKBUL	2	16	1	16	2
BLOCK EXTRACT	160	205	43	221	210
BLOCK EXPORT	160	18	43	38	210
BLOCK IMPORT	160	27	43	47	210
BLOCK APPLY	160	197	43	238	210
BLOCK CLEAN	283	100	278	102	278
TOTAL	160	563	43	662	210
INCWOA EXTRACT	15	905	15	916	28
INCWOA EXPORT	15	26	15	27	28
INCWOA IMPORT	15	35	15	36	28
INCWOA APPLY	15	183	15	194	28
INCWOA CLEAN	282	104	278	103	277
TOTAL	15	1253	15	1276	28
INCWA EXTRACT	15	425	15	420	28
INCWA EXPORT	15	26	15	27	28
INCWA IMPORT	15	35	15	36	28
INCWA APPLY	15	183	15	194	28
INCWA CLEAN	282	104	278	103	277
TOTAL	15	773	15	780	28
ALERT GENERATE	5	91	6	91	12
ALERT EXPORT	5	12	6	13	12
ALERT IMPORT	5	21	6	22	12
TOTAL	5	124	6	126	12

UPDATE FUNCTION	AVGTIME	STDTIME	AVGSIZE	STDSIZE
BULK EXTRACT	20.60	1.34	266	28.07
BULK EXPORT	43.20	3.56	266	28.07
BULK IMPORT	51	5.05	266	28.07
BULK APPLY	33.80	1.10	266	28.07
BULK CLEAN	104.60	1.82	266	28.07
TOTAL	253.20	7.63	266	28.07
BLOCK BLKINT	531.80	6.26	208.60	3.78
BLOCK NUMBLK			3.80	3.90
BLOCK BLKBUL	15.80	.45	1.40	.55
BLOCK EXTRACT	210.40	7.64	93.20	86.07
BLOCK EXPORT	24	10.75	93.20	86.07
BLOCK IMPORT	33	10.75	93.20	86.07
BLOCK APPLY	229.20	27.73	93.20	86.07
BLOCK CLEAN	102.60	2.41	275.60	6.73
TOTAL	615	44.49	93.20	86.07
INCWOA EXTRACT	891	41.47	16.20	6.91
INCWOA EXPORT	26	.71	16.20	6.91
INCWOA IMPORT	35	.71	16.20	6.91
INCWOA APPLY	183	7.91	16.20	6.91
INCWOA CLEAN	103.40	1.14	275.20	6.38
TOTAL	1238.40	47.70	16.20	6.91
INCWA EXTRACT	414	21.37	16.20	6.91
INCWA EXPORT	26	.71	16.20	6.91
INCWA IMPORT	35	.71	16.20	6.91
INCWA APPLY	183	7.91	16.20	6.91
INCWA CLEAN	103.40	1.14	275.20	6.38
TOTAL	761.40	21.43	16.20	6.91
ALERT GENERATE	90.60	.55	6.20	3.42
ALERT EXPORT	12.20	.45	6.20	3.42
ALERT IMPORT	21.20	.45	6.20	3.42
TOTAL	124	1.22	6.20	3.42