THE UNIVERSITY OF CALGARY

AN INVESTIGATION INTO MODELLING AND MANAGING UNCERTAINTY IN GEOGRAPHIC INFORMATION SYSTEMS

by

ROSS MILLER

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE IN ENGINEERING

DEPARTMENT OF SURVEYING ENGINEERING

CALGARY, ALBERTA

SEPTEMBER, 1991

ISBN 0-315-75273-4

THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "An Investigation Into Modelling And Managing Uncertainty In Geographic Information Systems" submitted by Ross Miller in partial fulfillment of the requirements for the degree of Master of Science.

M. A. Chapman

Supervisor, Dr. M.A. Chapman
Department of Surveying Engineering

J.A.R. Blais

Dr. J.A.R. Blais
Department of Surveying Engineering

I.K. Crain

Dr. I.K. Crain
Department of Surveying Engineering

Nigel Waters

Dr. N.M. Waters
Department of Geography

Date 18 October 1991

ABSTRACT

Geographic Information Systems (GIS) technology are becoming increasingly important in our society. Information derived from GIS and used in decision making can have far-reaching effects on the land, its people, and the environment. Uncertainty in GIS is a large and complex problem which bears directly on the quality and suitability of this information. Despite its importance, it remains a relatively poorly researched subject area.

This thesis is concerned with the development of a prototype computing environment, which will assist in further research of the problem of the propagation of uncertainty in GIS. The types of uncertainty present in GIS are identified in the context of a communication paradigm, which views GIS as a sequence of models which transform data from their source to the end product. Uncertainty is introduced into the data as a result of the modelling processes, and includes such issues as the accuracy of spatial and non-spatial data, the effect of GIS transformations on these data, and data quality issues of consistency, lineage and completeness. Methods for representing and managing uncertainty are examined. A number of requirements are then identified which are necessary for the prototype. The prototype comprises GIS software, a PROLOG rule base, several utility and translator programs, and a small data set for experimentation, which exhibits many aspects of uncertainty. A PROLOG program is developed which enables the determination of the lineage of data from their source to some final GIS product. The PROLOG database is extended to include a meta-interpreter which allows for different representations of uncertainty measures to be included with the data and transformation rules, and for the propagation of these through the lineage of a data set. The meta-interpreter is used in an experiment of establishing inter-

layer consistency, involving attribute data, positional uncertainty, lineage data at the feature level, and logical consistency. Topological and coordinate data can be included in the rule base, and access to operating system utilities and applications programs are possible.

The prototype meets the computing requirements for uncertainty as identified. Further research is recommended, particularly in the area of developing models of uncertainty for GIS data and transformations, and in methods of managing these.

ACKNOWLEDGEMENTS

The completion of this thesis would not have been possible without the support and encouragement of many people. I would like to express my gratitude to my supervisors, Dr. M.A. Chapman, Dr. G.D. Lodwick, and Dr. V.B. Robinson; to Dr. K. Frankich and Mr. D.C. Deans of the British Columbia Institute of Technology for helping to make my studies possible in the first place; and to my fellow graduate students in the Department of Surveying Engineering. I would also like to acknowledge the financial support provided by the Natural Sciences and Engineering Research Council.

TABLE OF CONTENTS

APPROVAL PAGE ii

ABSTRACT iii

ACKNOWLEDGEMENTS v

TABLE OF CONTENTS vi

LIST OF TABLES viii

LIST OF FIGURES ix

Chapter 1 Introduction 1
- 1.1 Emerging Importance of Geographic Information Systems 1
- 1.2 Development of Geographic Information Systems 1
- 1.3 Uncertainty in Geographic Information Systems 5
- 1.4 Research Objectives 6

Chapter 2 Uncertainty In Geographic Information Systems 8
- 2.1 Conceptual Framework 8
  - 2.1.1 Levels of Measurement 9
- 2.2 Types of Uncertainty in GIS 10
- 2.3 Data Quality 11
- 2.4 Management of Uncertainty - General Approaches 13
- 2.5 Models and Sources of Uncertainty in GIS 15
- 2.6 Summary 17

Chapter 3 Representing and Reasoning with Uncertainty in GIS 19
- 3.1 Modelling Uncertainty in the Geosciences and GIS 19
- 3.2 Information Theory 20
- 3.3 General Models of Reasoning about Uncertainty 23
  - 3.3.1 Bayes' Theorem 23
  - 3.3.2 The Mathematical Theory of Evidence 25
  - 3.3.3 Fuzzy Set Theory 28
- 3.4 Knowledge-based Approaches 30
- 3.5 Summary 32

Chapter 4 A Prototype Machine for Investigating Uncertainty 33
4.1 Requirements for the Prototype 33
4.2 Software 34
4.2.1 GIS Software 35
4.2.2 PROLOG 37
4.2.3 Other Languages 46
4.2.4 Operating System Software and Utilities 47
4.2.5 Procedures and Rule Base in UNCLE 49
4.3 Summary 49

Chapter 5 Database Development 50
5.1 Introduction 50
5.2 Study Area 50
5.3 Determination of Data Required and Data Sources 53
5.4 Data Preparation 56
5.4.1 Map Digitizing 56
5.4.2 Data Validation (cartographic data) 58
5.4.3 Data validation (attribute data) 67
5.5 Summary of GIS Database 67
5.6 Summary 68

Chapter 6 Experimentation 69
6.1 Lineage in Geographic Information Systems 69
6.2 ARC/INFO Architecture and Data Organization 70
6.3 Meta-data in ARC/INFO 75
6.4 Modelling Lineage in UNCLE 80
6.4.1 Source Documentation 80
6.4.2 Establishing the Lineage Database 81
6.4.3 Meta-Level Programs in UNCLE 92
6.5 Utilizing Spatial and Non-spatial Data in UNCLE 103
6.5.1 Direct Reading and Writing of ARC/INFO Files 104
6.5.2 File Translation Methods 106
6.5.3 Experimentation with Feature Level Data 107
6.6 Summary 119

Chapter 7 Conclusions and Recommendations 121
7.1 Conclusions 121
7.2 Recommendations 122

References 125

LIST OF TABLES

Table 3.1 After Garvey et al (1981) 27
Table 5.1 Map features prepared for study area. 56
Table 5.2 Summary of ARC/INFO coverages. 68
Table 6.1 Values of positional uncertainty used in overlay operation. All values in metres. 101

LIST OF FIGURES

Figure 4.1 A PROLOG program for spatial relationships. 39
Figure 5.1 Location of study area 51
Figure 5.2 Stages in preparation of data for entry into GIS 54
Figure 5.3 Text file produced by DIGIT, showing registration data and digitized coordinates. The ! character denotes a comment. 58
Figure 5.4 Content and structure of an ASCII chain file produced by ODYSSEY 62
Figure 5.5 PENELOPE error report for soils data, showing the conversion of LDG/LDB "spaghetti" to CDG/CDB chain topology. Tolerance values are reported, and chains and polygons can be identified by number. Distances in metres. 64
Figure 6.1 Organization of an ARC/INFO workspace, showing the workspace directory ('arcwork') and the coverage and 'info' directories and some common ARC/INFO files. 72
Figure 6.2 Partial INFO listing of workspace files. 74
Figure 6.3 ARC/INFO workspace and coverage log files, showing date and time, connect time, cpu usage, disk input/output (not applicable for Sun computers), and the ARC/INFO commands and parameters entered. 77
Figure 6.4 a) A watch file and b) its AML equivalent. 78
Figure 6.5 An ARCEDIT AUDITTRAIL, showing detailed transactions on individual features. 79
Figure 6.6 An UNCLE 'hist' file, showing original information from DIGIT program, and incorporating operator comments. These can be displayed from within ARC/INFO in a 'pop-up' screen. 82
Figure 6.7 PROLOG facts for documentation on source manuscripts. Text between '/*' and '*/' are comments. 83
Figure 6.8 PROLOG Facts for the 'soils' coverage, after conversion from ARC/INFO LOG files. Derived from the LOG file shown in Figure 6.3b. Commands become functors, and coverage and parameters become arguments. 83
Figure 6.9 A portion of the PROLOG rule base in UNCLE 85
Figure 6.10 An interactive session with UNCLE, showing how source documentation can be retrieved for an original coverage. 86
Figure 6.11 Example of a PROLOG query in UNCLE to determine the lineage across one or more 'generations' of coverages. 89
Figure 6.12 Determining lineage over several generations, with several source coverages. 91
Figure 6.13 Finding all of the descendants of a coverage. 92
Figure 6.14 A proof tree in UNCLE which shows the relationships between coverages and how they are related in the lineage. 94
Figure 6.15 Computing uncertainty in UNCLE. 97

Figure 6.16 Computing intervals of uncertainty. 98
Figure 6.17 Output from UNCLE showing positional uncertainty propagated during processing. 'CF' values in metres. 102
Figure 6.18 Polygons for 'soils' and 'lakes' coverages. Numbers shown indicate polygon identifiers. Enlargement of shaded study area shown in Figure 5.1. 108
Figure 6.19 Overlay of 'soils' and 'lakes' coverages. Enlargement of shaded area shown in Figure 5.1. 109
Figure 6.20 Automatic elimination of spurious polygons. Dotted lines indicate more accurate line work from 'lakes' which were eliminated. Enlarged from study area shown in Figure 5.1 112
Figure 6.21 Report from UNCLE listing definite inconsistencies, with lineage and uncertainty measures included. 116

Chapter 1

Introduction

1.1 Emerging Importance of Geographic Information Systems

Information Technology (IT) now permeates virtually every aspect of western society, affecting not only the everyday business of government, business and industry, but also the personal lives of individuals. An increasingly important and integral component of IT are computer-based Geographic Information Systems (GIS), which are beginning to have an equally wide ranging effect on our society. GIS have expanded greatly in use in recent years; properly implemented, managed and operated, they have tremendous beneficial potential in applications which require spatially referenced geographic data. Historically such spatial data were in the form of paper maps and paper-based records, which restricted efficient organizational use, distribution and associated costs (Lodwick, 1986). Computer-based GIS, on the other hand, have the potential to provide users with access to accurate, timely data and methods for retrieving and manipulating these data. They offer a means for rapidly producing high quality and up-to-date information in the form of maps and reports while at the same time achieving improved efficiency in data collection, distribution and usage.

1.2 Development of Geographic Information Systems

The advances in computer science and technology in the past decades which

underlie and make possible Information Technology - such as computer hardware and software, computer graphics, database management systems - have also facilitated the development of GIS. But GIS are only made possible by integrating with these technologies the results of equally significant advances in the theory and practice of computer cartography and digital mapping, geographic analysis, and measurement sciences, especially those in the surveying and mapping profession. During the last three decades a great deal of important research and development occurred in these areas, and continues today. Nevertheless, to a great extent the theoretical and conceptual developments in GIS have lagged behind the software and hardware innovations of the information revolution of the past two decades. Today, GIS remains largely technology driven, as demonstrated by the existence of over a hundred commercial software packages available on every type of computer, from microcomputers to mainframes while at the same time, some very fundamental concepts and principles relating to spatial data and their use have not been developed and implemented. For example, despite the fact that errors have always been present in measured and recorded geographic and cartographic data (Openshaw, 1989) there has been remarkably little research into error in GIS and its effect on the results of studies made (Burrough, 1986). Recently there has been increasing attention in research and development to the accuracy of spatial data bases and the propagation of error in GIS (Goodchild and Gopal, 1989). However, relatively little has been implemented in commercial systems to help the user deal with problems of error or accuracy. Apart from reporting on such processes as map

registration errors during digitizing, commercial GIS software offers little in terms of reporting on accuracy of data or transformations.

There are many reasons why it is important to have such information when using a GIS. A definition of GIS that is acceptable to all of the different types of users has still not been established. In this thesis the definition of GIS is based on Cowen (1988) and Hamilton and Williamson (1985). GIS is viewed as a decision support system, providing tools for legal, administrative and economic decision-making, and as an aid for planning and development. Data in a GIS are the raw material of information (Bédard 1986b). Regardless of the application area, the information derived from these GIS data should enable users to make more effective decisions regarding the natural, cultural and socio-economic environment in which these users live and work.

The application sectors for GIS are numerous and diverse, and as decision support systems in these areas the value of GIS-derived information is critical. For example, GIS are used in urban and regional planning and development, environmental monitoring, mapping, and resource evaluation and development, to name but a few. The number and variety of potential users, the magnitude and cost of the projects involved, the growing scarcity of land and natural resources, and the potential impact which decisions based on GIS technology have on business, the environment, and society, indicate the importance of having reliable information. As spatial data bases become larger and more long term, as they expand to serve a wider variety of users with different information needs, and as more intelligent and

sophisticated GIS are developed, the need for knowledge of the uncertainty of data and the reliability of the derived information will also increase. Producers and users need to know how good the information is, in order to make informed decisions and to reduce any liability and potential litigation. Recent literature on data quality and accuracy of spatial databases attest to the growing recognition and importance of this problem (Grady, 1988; Chrisman, 1983).

GIS are rapidly replacing traditional methods of handling spatial data for many types of users. Once relegated to mainframe computers, GIS software incorporating powerful analytical tools are now available on microcomputers and workstation platforms costing substantially less. Data are becoming more accessible directly in digital form from government and private vendors; and conversion of analog map data, though still a major cost in the development of GIS databases, is easily performed by such means as scanning, or manual digitizing. Commercial databases for storing non-spatial data are available, affordable, easy to use, and familiar to most people. These developments facilitate the acquisition and use of spatial data handling and analysis capabilities by both knowledgeable and inexperienced users alike. For users with a poor understanding of GIS processes and data accuracy issues, a GIS has as large a potential for misuse as it does for positive benefit. The availability, popularization and power of GIS technology provides a relatively easy means of generating information, and yet the fact that the quality of the information should determine the applications for which it is suitable is often neglected. If information was available through GIS software which indicated the

accuracy of the data and the processes applied to these, then the likelihood for misuse by end users could be reduced.

1.3 Uncertainty in Geographic Information Systems

In GIS, the sources of spatial data, the nature and volume of these data, the methods for acquiring, inputting, and modelling them, the processes they can undergo, and the ways in which they can be presented vary considerably. In the GIS database the data may be incomplete, out-of-date, or inconsistent. All of these factors can contribute to the existence of uncertainties in the database; in turn, these uncertainties affect the reliability of the information derived from GIS. Using this information in the context of a decision support system might require additional knowledge of the source of the data and how it was derived.

Generally speaking, uncertainty in GIS is a poorly understood and complex topic, yet a critical one to be addressed if GIS are to reach their potential. The problem of accuracy of spatial databases is one of the main research agenda items of the National Center for Geographic Information and Analysis (NCGIA, 1989).

In this thesis, the term "uncertainty" includes the common problems of accuracy and error in GIS software and databases referred to above. Most commonly, this term is used in a probabilistic sense. In surveying and mapping, for example, "positional uncertainty" is sometimes used synonymously with "positional accuracy" to refer to the correspondence of observed or adjusted points in a survey to their "true" values. Similarly in geological analysis, uncertainty takes on a probabilistic

interpretation (Davis, 1986). In geology and geography, observations of the spatial and non-spatial (attribute) characteristics of data are typically recorded on nominal, ordinal, ratio or interval scales of measurement. Uncertainties associated with these are also managed with statistical techniques. However, Robinson and Frank (1985) also note that a great deal of geographic data are described with linguistic terms and encoded in a subjective fashion, for example, those measured on nominal or ordinal scales. Ambiguity - uncertainty - is introduced as a result of this. Other terms used to describe uncertainty in such data are inexactness, imprecision and vagueness. These are essentially non-statistical in nature and some non-statistical approaches have been proposed to deal with these (Robinson and Frank, 1985; Robinson and Strahler, 1984).

In this thesis, "uncertainty" is therefore used in a broad sense and refers to the statistical and non-statistical aspects of the spatial and non-spatial attributes of data and their transformations. Additional aspects of dealing with uncertain data and situations where considerations of data completeness, currency, consistency, source and lineage are also considered as part of the problem of uncertainty. This is described more fully in Chapter 2.

1.4 Research Objectives

This thesis examines some of the issues relating to the complex problems associated with uncertainty in GIS. The main objective is to develop a prototype computing environment for further research and development into evaluating

uncertainty in GIS. This is approached by developing means to monitor uncertainty, using lineage trails of GIS data and transformations. The operational context is that of providing information on the quality and relationships of the data, so that this can be used to establish consistent GIS databases.

Prototypes are rapidly constructed yet working models of final systems. They allow for the discovery of problems, and help to insure that the final system meets user requirements. In this thesis, the prototype developed is a computer environment of hardware, software, data and procedures which allow for further investigation into uncertainty. This is achieved in the following manner.

A literature review on uncertainty is undertaken to identify the important concepts of uncertainty in GIS. This is the subject of Chapters Two and Three. Chapter Two describes a conceptual framework for investigating uncertainty and also gives examples of some of the sources and many problems associated with uncertainty in spatial data and geoprocessing in a computer-based GIS environment. Chapter Three examines some traditional and more recent models for representing and reasoning with uncertainty in spatial data. Based on the findings of Chapter Three, the functional requirements of the prototype, and the tools selected for it, are identified in Chapter Four. Chapter Five describes the development of the GIS database. Chapter Six describes the development, implementation and testing of the prototype with the database. Chapter Seven provides conclusions and recommendations for further research.

Chapter 2

Uncertainty In Geographic Information Systems

2.1 Conceptual Framework

The conceptual point of departure for this thesis is the "communication paradigm" of LIS[1] (Land Information Systems) first proposed by Bédard (1986a) and summarized in Bédard (1986b; 1987). Bédard's study of the nature of data in land information systems presents GIS as a complex communication process between the data collectors (for example land surveyors) and the eventual users of these data (decision makers). GIS consist of a sequence of physical and cognitive models which transmit information about real world geographic phenomena by means of a "technical subsystem" (the data processing functions of storage, retrieval, manipulation etc.). All aspects of a GIS can be explained in terms of this framework, and the focus is on the modelling, information and communication aspects involved in this process. Bédard identifies two aspects of the communication process which introduce uncertainty.

The first cause of uncertainty is related to the modelling processes inherent in GIS. Since models can only approximate reality, a lack of homomorphism exists between the geographic reality, and the cognitive model which a user builds from the retrieved information (Robinson and Frank, 1985; Bédard, 1987). Consequently,

---

[1] Bédard adopted the term 'LIS' for his research. In this thesis, 'GIS' and 'LIS' are treated as synonymous, and for the sake of consistency, 'GIS' will be used henceforth.

although the ultimate goal of GIS might be to deliver "perfect information" (i.e. complete homomorphism) in fact the processes required in modelling reality introduce uncertainties in the data at every step in a GIS. These modelling limitations may be related to fuzziness in the identification or labelling of entities, or in limitations in the measurement of properties of these entities (Bédard, 1987). These are discussed more fully below.

The second source of uncertainty is related to human factors. The human participants of GIS have limited capacity as information processors and may introduce subjectivity in data. For example, one's frame of reference (i.e. world view) and tendency to "satisfice" may directly affect the reliability of data in a GIS (Bédard, 1987).

2.1.1 Levels of Measurement

The measurement or description (identification, labelling) of geographic phenomena may fall into one of several levels or scales of measurement (Campbell, 1984; Robinson et al, 1984).

Nominal level In this level, phenomena are separated into categories or type without any reference to order or ranking. The description refers to the existence (or non-existence) of data at a particular location and its qualitative, rather than its quantitative, characteristics. For example, land may be classified as agricultural or industrial.

Ordinal level This may be described as nominal with ranking. Some indication of

relative magnitude of quantitative characteristics is provided, yet without any expression of exact quantitative measure. For example, agricultural land might be classified as low, medium or high productivity; industrial land might be classified as light or heavy; terrain might be classified as steep, moderate or gentle slope.

Interval and ratio scales Both of these levels involve quantitative measurements to describe the phenomena. The distinguishing characteristic of ratio scales is an absolute (non-arbitrary) starting point, and a constant distance between increments. For example, a ratio scale might be used to measure agricultural production in kilograms per hectare, with a starting point of zero (i.e. no production) and therefore meaningful comparisons can be made between values on the scale (100 kg/ha is exactly twice that of 50 kg/ha). On the other hand, in interval scales, an arbitrary starting point may be chosen, but relationships between values on the scale are not easily comparable. In the Celsius temperature scale, 0 degrees, the freezing point of water is arbitrary in so far as it does not really mean the absence of temperature, and one cannot say that 100 degrees is twice as warm as 50 degrees. By far, most of the phenomena modelled in GIS are measured on a nominal, ordinal or ratio basis.

2.2 Types of Uncertainty in GIS

Four types or levels of uncertainty can be identified in GIS (Bédard, 1987). The first is a conceptual uncertainty, (e.g. fuzziness in identification) which affects the classification of phenomena. An example would be the classification of continuous or naturally variable phenomena into discrete taxonomic or spatial groups. The

second and third types of uncertainty are related to measurement limitations of the quantitative and qualitative properties, respectively, of an entity (Robinson and Frank, 1985). These give rise to uncertainties in the spatial and non-spatial attributes of the phenomena. Thus there may be locational (positional) uncertainty, which refers to inaccuracy in quantitative values and fuzziness in qualitative values used for location in space and time of an entity, i.e. its spatial attributes. For non-spatial attributes, there is descriptive uncertainty, which is uncertainty in the attribute values of an entity, either imprecision in quantitative values (e.g. degree of slope, percentage of pine) or fuzziness in qualitative values (e.g. soil characteristics). Types two and three uncertainty are obviously closely tied to the levels of measurement discussed above.

The fourth type of uncertainty is meta-uncertainty, which refers to the degree to which the other types of uncertainties are known. This type can also be described in terms of the levels of measurement. Altogether, these four types combine to produce the total amount of uncertainty in a GIS (Bédard, 1987).

2.3 Data Quality

The types of uncertainties defined by Bédard provide a relatively complete framework for studying the problem of uncertainty. Nevertheless it is useful to review other studies relating to this problem and examine the terminology used. Literature on the problem of data quality in geographic databases presents another aspect of the problem. Chrisman (1983) states that GIS databases must include more than just

spatial data; they should include information on how this data is known. This will serve not only application end users but producers of digital data. The term adopted to refer to this is data quality and a data quality report for users would include all of the following components:

1) lineage, describing the source material, how data were derived, including such information as transformations employed, dates of updates;
2) positional accuracy, giving the quality of control surveys used, and based on established geodetic standard procedures;
3) attribute accuracy, giving a numerical estimate of expected discrepancies of non-spatial characteristics, in a manner similar to positional accuracy measures;
4) logical consistency, describing the integrity of relationships of the internal data structures. In a vector topological database this includes geometric and topological tests of features; and
5) completeness, describing the exhaustiveness of spatial and taxonomic properties, the consistency with which features have been assigned, and the selection criteria (including such values as geometric tolerances) and standards employed

(Chrisman, 1983; NCDCDS, 1988).

It is obvious that the concerns of "data quality" and "uncertainty" are much the same. For example, types 2 and 3 uncertainty as described by Bédard and data quality items 2) and 3) as listed above, are concerned with positional and attribute accuracy. Also, the data quality components of lineage, completeness and logical consistency as described above can serve to extend the notion of meta-uncertainty as

defined by Bédard, by giving additional information on the degree to which the other types of uncertainty are known. In this thesis, therefore, these are included as part of meta-uncertainty or meta-data. These might be represented as descriptive accounts of data which use a variety of levels of measurement.

In this thesis this broad description of uncertainty is used as the basis for identifying the general requirements for the prototype. However, the experimental component of this research deals primarily with types 2 and 4 uncertainty. The type 2 uncertainties investigated are the positional accuracies of digitized and processed map features in a GIS database, where accuracy is a measure of how close the digital representation of the position is to the "true" value. This can also be viewed as the errors in the lines which depict the features. Associated with the digital line representations are type 4 uncertainties, which describe the lineage, transformations, and inconsistencies of the database.

2.4 Management of Uncertainty - General Approaches

Bédard (1987) identified two means to deal with the uncertainty in GIS; these have been referred to as uncertainty reduction and uncertainty absorption. Uncertainty reduction refers to decreasing the fuzziness associated with identification of an entity, or increasing the accuracy and precision in the description or location of the entity. This can be accomplished by improved technical or procedural methods. For example, in surveying, improved measurement techniques and the use of mathematics to adjust repeated observations may be employed. These might be used,

for example, to obtain a better estimate for the position of a point. For qualitative data the use of standard terminology in classification and fuzzy logic might be applied (Bédard, 1987). These are equivalent to improving some aspect of the modelling process. These efforts notwithstanding, there will always remain a certain amount of uncertainty present in the GIS database.

Uncertainty absorption occurs when either the model-maker or user of geographic information absorbs the effect of the remaining uncertainty in the data. The amount of absorption can be viewed as the level of risk entailed by using or providing uncertain data. Institutional absorption of uncertainty (e.g. guaranteeing title and boundaries with indemnity funds) in effect guarantees the certainty of the data, even though the model may not exactly reflect the underlying geographic reality (e.g. a legal plan with prorated dimensions for lot lines forms part of the final model for the landowner, when in fact the 'post in place' indicates the true extent of the parcel). This institutionalized absorption of uncertainty and guarantee of data is not common in GIS. More typically, the data in a GIS have varying amounts and types of uncertainty, and uncertainty reduction by technical and procedural means appears to have received more attention. This is evidenced by such efforts as improving digitizer accuracy by improving the digitizer hardware, by evaluation and selection of the best transformation for digitizing (Petersohn and Vonderohe, 1982), or by introducing quality control procedures. When uncertainty cannot be reduced further and the data are not guaranteed, the user absorbs the remaining uncertainty as an element of risk in decision making. Two types of absorption can be identified. In the

first, no indication of the reliability of the data is given (indeed the user may actually receive information from GIS service companies with legal disclaimers attached). In the second, some measure of data quality is given or may be determinable, but determination of the suitability of the information is still left up to the user. In either case, to further reduce uncertainty to a satisfactory level, alternative methods, such as field checking to verify positions, may be employed. Although users will ultimately determine the appropriateness of the information they receive and make their decisions based on it, it is desirable that the producers be responsible for supplying additional information on the quality of the data (Grady, 1988).

2.5 Models and Sources of Uncertainty in GIS

The many modelling processes in GIS provide numerous examples of situations where uncertainty can arise. Some examples of these are now given, classified in terms of the technical subsystems of GIS.

Data acquisition and input

In data acquisition and input, well established mathematical models (least squares adjustment of survey networks, photogrammetric block adjustments) are employed to provide better estimates of field observations. These typically form the basis for digital database creation, and the positional uncertainties generated at this stage, though perhaps relatively small compared to other processes, nevertheless may affect all subsequent analyses. The choice of the best transformation algorithm for manual digitizing (Petersohn and Vonderohe, 1982) is another example of the use of

mathematical models which contribute to uncertainties. In contrast, a soil scientist performing a field survey and classification according to the terminology of a standardized taxonomy, or a photo interpreter doing forest cover classification by linking spectral signatures to information classes, employs different types of physical and cognitive models during the inputting of data, and may introduce additional subjective elements (Robinove, 1981).

Data storage and retrieval

A geographic database designer employs models to express the relationships among spatial entities, and encodes these in a database of a particular model (e.g. relational or network), which may have limited expressive power in terms of the spatial, taxonomic or topological characteristics. For example, the relational model may not support 'one to many' and 'many to many' relationships, which may be required to model the geographic phenomena for a given purpose. This will result in the need for additional techniques and storage which lead to more indirect and complex methods of modelling, and which may in turn introduce uncertainty.

The computer hardware and software configuration in which spatial data are stored is itself a model, with intrinsic limitations such as finite machine precision, such as the support of single precision numeric values only which result in rounding errors.

Data manipulation

In retrieval and processing of data, algorithm design for vector-raster conversions, automated generalization of linework, or geometric transformations such as scaling

or rotation may be performed, providing further modelling of the original data, and increasing uncertainty. Franklin (1984), for example, has demonstrated the implications for topological integrity of points and polygons by applying scaling and rotation transformations to integer values; the positions of certain points were changed from being inside a polygon to outside a polygon by the transformations.

In polygon processing to build topology or overlay two sets of polygons, software system defaults for geometric tolerances which automatically coalesce points may eliminate points without a user being aware.

Data output

Cartographic output is likewise affected by such factors as internal storage precision and algorithm design. For example, at certain view scales in a graphics program, there may be graphic misclosures of connecting linework as a result of a line intersection algorithm computing an intersection point which does not lie exactly on the digital representation of the lines. Some software might apply generalization algorithms to lines and produce highly generalized lines or curves at large view scales.

2.6 Summary

A conceptual model of uncertainty is presented based on Bédard's communication paradigm. Four levels of uncertainty are described. Uncertainty may be conceptual, related to measurement limitations of qualitative and quantitative characteristics, or may be meta-uncertainty (meta-data). Meta-uncertainty is extended to include the

requirements of data quality. The description of phenomena and their associated uncertainty is based on levels of measurement as identified in the cartographic literature. General strategies for managing uncertainty - uncertainty reduction and uncertainty absorption - are described. Examples of modelling processes and the kinds of uncertainty they can generate are described in terms of the data processing functions (technical subsystems) of GIS.

Chapter 3

Representing and Reasoning with Uncertainty in GIS

This chapter provides an overview of research directions in uncertainty from the fields of surveying and mapping, communication theory, artificial intelligence (AI), expert systems (ES), and geographic information systems in order to identify how the different types of uncertainty previously described may be represented and reasoned with in GIS.

3.1 Modelling Uncertainty in the Geosciences and GIS

As noted in Chapter 1, the development of uncertainty modelling and error propagation in GIS lags behind other theoretical and technological aspects. However, in the enabling disciplines of GIS, (surveying and mapping, geography, geology, soil science), there is a strong research record in such areas of uncertainty as least squares adjustment, transformation analysis, statistical spatial analysis, and surface interpolation. Generally speaking, this research has been concerned with the quantitative and qualitative characteristics of geographic phenomena (types 2 and 3 uncertainty). The approach has been mainly statistical, involving the formulation of models, including associated error terms, the application of these to observed phenomena, and inferencing from the results. These have been essential in the development of the disciplines themselves and GIS.

However, the inclusion of these models in GIS by vendors has been limited,

and GIS software also limits the ability of users to add these to the software. Apart from the ability to determine such errors as those associated with registration of a map to a digitizing tablet, GIS software rarely reports any type of statistical error during processing. In some cases, it may be possible to determine from documentation the type of algorithm applied in geoprocessing operations (eg. rubber sheeting may use, for example, either affine or projective algorithms, line simplification algorithms may be identified) but not how it was implemented. This is unfortunate, though understandable, given the proprietary nature of commercial systems.

Typically, research in the disciplines mentioned implements the models using stand-alone, specialized computer algorithms in high level code accessing flat file data. The facility to add these to GIS as user defined routines is limited. This is likewise a function of proprietary systems, especially where there is limited, if any, access to underlying data structures, and the absence or rudimentary state of programming or macro programming capabilities in commercial software. In order to understand and utilize fully these types of uncertainty in GIS, it is necessary to have this facility.

3.2 Information Theory

Information theory provides the theoretical basis for Bédard's communication paradigm for GIS. The field of information theory provides a mathematical definition of uncertainty. Information theory has its origins in a theory of communication which

studied the transmission of electronic signals from some source to some recipient by means of some channel. The information transmitted is expressed in terms of the probabilities of the signals, and its measurement represents an average of the information system (Shannon and Weaver, 1959; Meadow, 1973). The measure of information proposed by Shannon is based on the probability of selection of each of the decision alternatives facing an information source. The amount of information, H[p], is given by the equation

$$H[p] = -k \sum_{i-1}^{n} p_i \log p_i \qquad (1)$$

where p_i represents the probability of the signal occurring. If the probabilities of signals occurring is 0 or 1, then no information results; these two extremes represent states of certainty. On the other hand, if the probabilities of the signals lie between 0 and 1, then some amount of uncertainty is present. This demonstrates the notion that information is the dispelling of uncertainty, and that if there is no uncertainty, there can be no information (Meadow, 1973).

Although Shannon provides a measure of information (or uncertainty) he does not define it. His concern is the technical or engineering aspect of information transmission, and does not include semantic and pragmatic aspects of the information (Meadow, 1973; Blais, 1991). Information theory and other information measures have been used in digital image processing and pattern recognition (Blais and Boulianne, 1988) and it appears to be valuable in studying complex systems (Klir, 1987). Information theoretic concepts have been used to a limited degree in

cartographic communication theory, where they have been useful in identifying stages in the communication process, but they have failed to provide much insight into the map reading process (Head, 1984). Since maps are a primary information product of GIS, this may also be applicable to GIS. However, little research has been done with respect to the application of information theory in complex information systems such as GIS. Head's conclusions notwithstanding, further research is required to determine the role of information theory in dealing with uncertainty in GIS. Information theory as it is applied to digital images may be able to provide measures of uncertainty which can be utilized in conjunction with GIS. For example, there is a strong trend in GIS towards integration of raster and vector data and, in particular, the use of digital images. Digital imagery acquired from satellite remote sensing, and digital orthophotography from aerial surveys, are playing an increasingly important role in providing data for more conventional raster- and vector-based GIS. These can be used, for example, in such applications as environmental monitoring, land use change, and map (database) revision. Automated methods for carrying out these will require digital image processing and pattern matching capabilities, which will depend on the results of further research in information theory (Blais and Boulianne, 1988). Blais (1991) points out that information theory has unlimited potential in spatial information processing, and suggests that it can play a role in quantifying information in terms of ambiguity, fuzziness, and similar types of uncertainty.

3.3 General Models of Reasoning about Uncertainty

This section describes three theories of modelling uncertainty using numerical measures, and reasoning about the uncertainty. These theories in particular have received considerable attention and been examined for their suitability in representing and reasoning with different types of uncertainty. They are Bayes' theorem, the mathematical theory of evidence, and fuzzy set theory. Although they do not exhaust the possibilities, they are representative of statistical and non-statistical approaches, and demonstrate some additional requirements for modelling and managing the different types of uncertainty in the Bédard paradigm.

3.3.1 Bayes' Theorem

Bayes' theorem is based on fundamental probability theory, and therefore works best in dealing with uncertainty about facts due to randomness or variability rather than imprecision or incompleteness (Stoms, 1987). It is therefore suitable for types 2 and 3 uncertainty where the quantitative or qualitative characteristics have a statistical basis. Uncertain hypotheses (e.g. the classification of a pixel as crop type A, B, or C) have associated probabilities between zero and one and the sum of all probabilities for hypotheses is one. Bayes' rule uses conditional probability to predict or update the probability of an hypothesis by combining the prior probability of the event, the probability of evidence for the event occurring, and the likelihood of the evidence given that the hypothesis is true. Equation 2 gives a form of Bayes' Theorem.

$$P(h \mid e) = \frac{P(e \mid h)\,P(h)}{P(e)} \qquad (2)$$

where

$P(h)$ *is the prior probability,*
$P(e \mid h)$ *is the likelihood of the event,*
$P(e)$ *is the probability of one's evidence, and*
$P(h \mid e)$ *is the posterior probability.* (3)

Equation 2 can be extended to handle multiple competing hypotheses and several pieces of evidence, as shown in Equation 3. This raises serious pragmatic issues, since it is required to know all possible combinations of all possible hypotheses, and is therefore frequently unworkable.

$$P(h_i \mid e_1, \ldots, e_m) = \frac{P(e_1, \ldots, e_m)\,P(h_i)}{\sum_{j=1}^{n} P(h_j)\,P(e_1, \ldots, e_m \mid h_j)} \qquad (3)$$

Several assumptions of Bayesian inferencing have been questioned. The theory assumes that a predefined and uniform distribution of values is known, and that probabilities can be assigned with precision. The commitment of partial belief to an hypothesis commits the remaining belief to its negation or alternative hypotheses and thus does not distinguish between uncertainty and ignorance. The use of conditional probability, which assumes mutually exclusive and exhaustive hypotheses and conditional independence of evidence, is not always valid (Cohen, 1985). Bayesian

inferencing has been used in remote sensing applications, pattern recognition and feature extraction. Shine (1985) makes clear that Bayes' theorem has met with limited success in such areas, and suggests that other approaches may provide a better solution. Bayes' theorem has also been employed to study the accumulation of error in thematic raster overlay (Newcomer and Szajgin, 1984).

3.3.2 The Mathematical Theory of Evidence

The mathematical theory of evidence (Shafer, 1976) differs from Bayes' theorem in several important respects. It allows degrees of belief in subsets of hypotheses. In this theory, a frame of discernment, Θ, is the set of all propositions about the exclusive and exhaustive possibilities in a domain (Barnett, 1981). This frame of discernment, and subsets of it, are given basic probability assignments, denoted as *m*, with values between 0 and 1. Therefore Θ has a potential maximum value of a probability of 1. Unlike Bayes' theorem, probability assignments for the subsets of hypotheses need not sum to one. When assignments are made to subsets, the *m* value assigned to Θ drops by a corresponding amount. At any given time, the *m* value of Θ represents what is not known about the situation, and this allows a degree of doubt or ignorance to exist, and allows other probability assignments to change when new evidence becomes available. Belief functions - probability assignments - are therefore a measure of the chance that the evidence demonstrates the truth of an hypothesis (Stoms, 1987). A belief function measures the probability that the evidence implies some hypothesis and is a lower bound on the probability

of the truth of that hypothesis. This lower bound is sometimes termed the support or necessity for that hypothesis. An upper bound known as a plausibility measures the degree to which the evidence fails to refute the hypothesis. This gives rise to the concept of the evidential interval, i.e. the difference between the support and the plausibility of a hypothesis. Propositions about hypotheses are represented as shown below:

$$A_{[s(A),p(A)]}$$

where

A	*is the proposition,*
s(A)	*is the support for A, and*
p(A)	*is the plausibility of A.*

Table 3.1 demonstrates some important points about how uncertainty is represented using this method. When the support and plausibility measures are equal, evidential reasoning produces the same results as Bayesian probability. Stoms (1987) gives the following practical example of how these intervals are computed. Assume that the support for causes of sub-optimal biomass of crops are determined to be

s(a) = .25, that the crops are water-stressed,

s(b) = .15, that the crops are nutrient-stressed,

s(c) = .40, that the crops are insect-stressed, and

s(d) = .20, that the cause is unknown.

The plausibilities of these causes can be determined as either 1 minus the sum of the support of the other known causes, or as the support for that cause plus the distributed support, s(d).

Therefore,

$$p(a) = 1 - (.15 + .40) = .45,$$

$$p(b) = 1 - (.25 + .40) = .35, \text{ and}$$

$$p(c) = 1 - (.15 + .25) = .6$$

and the evidential interval for cause a, for example, is [.25,.45].

$A_{[0,1]}$	no support; full plausibility; no knowledge at all about A
$A_{[0,0]}$	no support; not plausible; A is false
$A_{[1,1]}$	full support; no evidence to contrary A is true
$A_{[.25,1]}$	some support and plausibility; evidence provides partial support for A
$A_{[0,.85]}$	no support, some plausibility; evidence provides partial support for not A
$A_{[.25,.85]}$	probability of A is between .28 and .85; evidence simultaneously provides support for both A and not A

Table 3.1 After Garvey et al (1981)

The theory of evidence may help address types of uncertainty which deal in incompleteness (type 4 uncertainty) and which may rely on further and possibly independent sources of new information. However, the theory relies on assumptions of independence of evidence and exclusive propositions; these cannot always be justified (Barnett, 1981). In the context of GIS, where the integration of data from diverse sources is a problem, this inferencing technique may have merit. It has been

employed in tactical military planning research (Garvey, 1987) and in classifying multispectral scanner data from multiple sources (Lee et al, 1987b). A good exposition of the mathematics involved in this theory is presented in Garvey et al (1981).

3.3.3 Fuzzy Set Theory

The third approach to uncertainty is fuzzy set theory (Zadeh, 1965). Fuzzy set theory involves uncertainty in determining whether an individual of some universal set belongs in a given subset, and expresses how much it is a member of such a subset. It is well suited to handle uncertainty of an imprecise or vague type, such as the linguistic ambiguities of natural language (eg. 'steep', 'moderate', or 'flat' slopes), and is applicable where the boundaries between sets are not 'crisp' or well defined. Examples in geography are those where the phenomena are gradually changing, such as climate, soil, or vegetation areas. Thus, fuzzy sets appear to be able to handle types 1, 2, and 3 uncertainty, where the concept or measurement of either quantitative or qualitative characteristics involve an interpretation of language (Robinson and Frank, 1985). Whereas classical set theory is based on Boolean logic and utilizes characteristic functions to determine if elements are members of a set or not, fuzzy set theory introduces the notion of grades of membership. Membership functions map elements into fuzzy sets. These membership functions produce grades of membership, which express the degree of compatibility of an entity with the concept of a given subset and measure the correlation between the linguistic values

and some numeric base values. A fuzzy subset A of a universal set X is defined by the function

$$\mu_A : X \rightarrow [0,1]$$

where

$$\mu_A(x) \textit{ expresses the grade of membership of } x \textit{ in } A.$$

(Klir, 1987). Grades of membership therefore assume values between 0 and 1. For example, if terrain slope is a linguistic variable on an ordinal scale, it could be associated with linguistic values such as 'steep', 'moderate', or 'flat'. These values may in turn be modified by such terms as 'very'. Given base values such as 10, 20 or 30 degree slopes, a 30 degree slope may be assigned a high grade of membership (e.g. 0.8) in the fuzzy set 'steep terrain', and a very low membership value (e.g. 0.1) in the set 'flat terrain' (Shine, 1985).

Fuzzy set theory subsumes classical set theory in so far as membership functions can also be used to produce the binary results of Boolean logic. For example, if the grade of membership for an entity in a set is 1, it is definitely a member of the set; if the grade of membership is 0, it is definitely not a member of the set.

A number of fuzzy set operations, such as union, intersection, complement, projection and join are possible, which are suitable for many types of geoprocessing operations in GIS. Leung (1987) has applied fuzzy set theory in a study of the

imprecision of climatic boundaries. He has identified the concepts of core (a non-fuzzy zone), boundary (a zone of transition) and edge (the outermost extremity) of geographic areas, and therefore demonstrated that fuzzy concepts can model the gradual transition and overlap between areas. Other research includes the application of fuzzy set theory to relational databases of geographic data and its implications for design of such systems (Robinson and Strahler, 1984) and to the problem of mixed pixel classification (Robinson and Thongs, 1985).

The inference methods outlined above are only three examples of approaches to managing uncertainty. Each has its own merits. There is considerable ongoing debate about the theoretical strengths of each of these and their application (Lee et al, 1987; Dubois and Prade, 1987a). Meanwhile research efforts aimed at alternative, non-numeric approaches such as models of endorsement (Cohen, 1985), managing additional qualitative aspects (such as non-monotonic logic), integrative approaches such as support logic programming (Baldwin, 1986), and general approaches (Yager, 1986a; 1986b) are being pursued. It is beyond the scope of this thesis to identify which of these approaches is the best, or which if any will dominate. Nevertheless, it is apparent by the nature of the types of uncertainty present in spatial data that these all have relevance and should be researched.

3.4 Knowledge-based Approaches

Managing uncertainty in GIS often requires the application of human expertise to satisfactorily reduce it for a given end use. For example, it is common

for an operator to use the evidence of proof plots and error reports, and then, using expert knowledge (meta-data) of the situation - for example, accuracy of data, job priority, end use application - make decisions concerning the accuracy, quality and suitability of the information. This suggests that intelligence is required to resolve some of these cases and that the application context may affect the choice of a solution. This is an example of a knowledge based heuristic approach to problem solving, i.e. using problem specific information to reach a solution.

One sub-discipline of the field of Artificial Intelligence (AI) is expert systems (ES), which are computer programs which advise on or solve real world problems (Robinson and Frank, 1987). ES often utilize a rule-based approach, implemented using programming languages such as LISP or PROLOG. The rule base consists of facts and rules which encode the domain specific knowledge of the expert. An inference engine acts on this rule base and delivers solutions via a user interface (Bratko, 1990). Some of the inference methods discussed above have been implemented in some sort of rule base.

The suitability of AI and ES has been researched for some of the problematic areas of GIS and computer cartography, such as map generalization and name placement (Robinson and Frank, 1987). Fisher (1989) contends that knowledge-based approaches to reliability in GIS data and its products should be included as part of a research agenda in GIS.

3.5 Summary

An overview of research in modelling and managing uncertainty is given. A number of models, and the type of uncertainty with which they deal, are described. Currently, there appears to be no single method of modelling and managing the four types of uncertainty identified by Bédard, though a great deal of theoretical research is being conducted to study various models. Based on the results of this review, it is possible to identify the requirements for further empirical research into uncertainty in GIS, which is the subject of Chapter 4.

Chapter 4

A Prototype Machine for Investigating Uncertainty

This chapter begins an investigation into the implementation of modelling the four types of uncertainty in GIS software, using the previous discussions as guidelines. A prototype computing environment is proposed which will support further research. The requirements for the prototype, the software and hardware selected, and the reasons for these, are described. The prototype facilitates the exploration of uncertainty, including lineage; it is therefore called UNCLE, for **UNC**ertainty and **L**ineage **E**xplorer.

4.1 Requirements for the Prototype

UNCLE consists of a hardware and software configuration, and a set of data for experimentation. Chapter 5 describes the development of the data set and the facilities and procedures used to do this. The research and programs developed in Chapter Five have proved important for the development of the prototype, as much of the knowledge gained was used for later developments. UNCLE is implemented on a SUN Microsystems SPARCstation 1, configured with 8 megabytes of random access memory.

The most important component of UNCLE is the software. Based on the findings of Chapter 3, this software must meet the following requirements;

Requirement 1. Provide access to the qualitative (non-spatial) and quantitative

(spatial) definitions of the geographic phenomena. This is required, for example, to perform conventional statistical analyses on the database.

Requirement 2. Allow for the addition of uncertainty measures and meta-data pertaining to Requirement 1.

Requirement 3. Capability to perform numerical computations using data from Requirements 1 and 2.

Requirement 4. Capability to include user programs, or the provision of sophisticated high-level programming, or both, in order to achieve Requirement 3.

Requirement 5. Capability to perform symbolic computations for knowledge- or rule-based applications.

Requirement 6. Ability to include rule-based programs for reasoning about uncertainty, in order to achieve 5.

In order to meet these requirements, a rule-based, coupled-systems approach is adopted. A coupled-system is any system which links both numeric and symbolic processing (Kitzmiller and Kowalik, 1987). These are sometimes called hybrid systems. The requirements listed suggest that both procedural languages, for numeric computations, and high-level languages such as PROLOG, for symbolic programming, might need to be employed.

4.2 Software

The software used in UNCLE can be divided into four categories;

1) commercial GIS software,

2) software development tools,

3) operating system software and utilities,

4) the procedures and rule base used for modelling and managing uncertainty.

4.2.1 GIS Software

The GIS software chosen for UNCLE is ARC/INFO, developed by Environmental Systems Research Institute of Redlands, California. It was selected because it provides an open, modifiable environment and appears to satisfy the above Requirements 1, 2, 3 and 4. ARC/INFO is a general purpose GIS software package, based on a topological vector data model. It provides a set of modules for entering, editing, querying, analyzing and displaying spatial and non-spatial data. Based on a "toolkit" approach to GIS, it allows users to build, from a set of basic tools provided through the modules, specialized and sophisticated procedures.

Users can give commands to ARC/INFO by means of command line sequences, or can develop, by means of Arc Macro Language (AML), customized routines and menu systems for command entry. AML also provides a high level programming language, with support for local and global variables, file and terminal input and output, conditional and unconditional looping and branching, mathematical functions, and string manipulation, as well as specific functions for accessing some parts of the ARC/INFO data structures and file system. Some AML "directives" allow for accessing operating systems functions, including other programs. AML provides a powerful tool for customizing ARC/INFO and helps meet Requirements

1, 2, 3 and 4. It is deficient however in supplying internal data structures such as numeric arrays of data, and has only rudimentary file input and output capability.

ARC/INFO is based in part on the ODYSSEY GIS program described in Chapter 5. For example, the polygon overlay algorithms and use of tolerances in topology building are very similar in both programs. In addition, the topological data model and underlying binary data structures used in ARC/INFO appear to be based on the ODYSSEY approach. This is particularly important because the data structures are not documented in the ARC/INFO manuals, however, they are well documented in ODYSSEY manuals. Although they are not identical this information has made possible the development of algorithms to read directly the ARC/INFO binary spatial files, and therefore help meet part of Requirement 1. These spatial files can be accessed more easily by translating them to a generic ASCII format (ARC/INFO "ungenerate" format). Unfortunately, this format produces only "spaghetti" data, and all topological information is lost in the translation. The spatial data can also be translated to such industry or government standard formats as AUTOCAD DXF or the United States Geological Survey Digital Line Graph (DLG) format. The latter standard produces an ASCII file, which is documented in government publications, and retains the topology of the original ARC/INFO data.

The non-spatial files in ARC/INFO (i.e. INFO database files, including ARC/INFO files and user attribute files) are also stored in binary format. These can be converted to ASCII format, or can be read directly and thereby meeting the other part of Requirement 1.

ARC/INFO provides access to a certain amount of meta-data about the processes and states of geographic features in its database, and therefore helps meet Requirement 2. This is essential to the development of the prototype, as it allows for the development of a lineage record of the data. This is discussed in more detail in Chapter 6.

4.2.2 PROLOG

The high level language PROLOG (for "programming in logic") was selected to develop a rule base for UNCLE. PROLOG is used extensively in Europe and Japan in Artificial Intelligence research and development. It is a powerful language for symbolic computation, and has been described as a relational database language. It is highly flexible, modular and recursive in nature. PROLOG differs from other high level languages in its declarative as opposed to a procedural approach to developing software. In most high level languages, such as FORTRAN or C, the programmer must concentrate on how to solve the problem, and code the step by step procedures to do so. In contrast, PROLOG has few constructs for controlling the flow of program execution. In PROLOG, the emphasis is on what is to be solved - by declaring goals to the PROLOG interpreter - and letting the interpreter find the solution. Typically, this involves an exhaustive search of the entire PROLOG database in order to come up with a solution to a goal ("query"). In this manner the user is freed from concerns on directing the program to find a solution. This obviously has implications for program efficiency; however, in prototyping, where the

main aim is to quickly obtain a working system, this is a secondary consideration.

The following discussion introduces some important terminology and concepts of PROLOG. It also demonstrates how a PROLOG program can be used and modified. These terms, concepts and operations are referred to again in Chapter 6 in the development of the prototype. It is therefore useful to have an understanding of these and how these are used. The discussion also demonstrates, in a practical fashion, why PROLOG is well suited for use in the prototype.

PROLOG programs can be thought of as databases consisting of rules and facts entered by the user. Rules and facts are contained in terms or clauses which have a head and a tail or body. If the body is empty the clause is a fact. If not, it is a rule and contains additional facts; these form a list of goals to be satisfied. The goals are delimited with commas which act as conjunctions. Disjunctions are most commonly represented by separate rules or facts. Clauses with no body are sometimes called predicates or relations and consist of a functor, and a number (arity) of arguments. The programmer assigns his or her own functors, and decides on the arity to suit the purpose at hand. Facts are represented by such statements as

```
father(john,paul).
```

This can be interpreted as 'john is the father of paul'. An example of a fact in the spatial domain might be

```
line(a,b).
```

If 'a' and 'b' represented labels for data points, this fact can be interpreted as defining the relation 'a is on a line with b'. An example of a rule using this fact is

connect(X,Y) :- line(X,Y).

Here the upper case letters denote variables, and the ':-' symbol is used to represent the operator 'if', which separates the head from the tail. This rule states that 'a point X connects to a point Y if X is on a line with Y.'

Figure 4.1 gives an example of a simple PROLOG program.

```
line(a,b).
line(b,c).
line(c,d).
line(e,f).
line(p,b).

connect(X,Y):-line(X,Y).
connect(X,Z):-line(X,Y),connect(Y,Z).
```

Figure 4.1 A PROLOG program for spatial relationships.

The user "executes" a program by submitting a question or goal to PROLOG. To submit a goal to PROLOG on the example database, the user might type the goal

?- line(a,b).

and PROLOG would respond by searching its database for any facts which satisfy this goal. This is done by pattern matching the users' question - its functor, arity, and arguments - against those stored in the database. In this case, PROLOG would respond by matching the users' functor 'line' against its database functors, and then search for any corresponding facts in the database which matched the arguments in brackets. In this example, it would find an exact match (the first fact in Figure 4.1) and respond to the user

Yes

indicating that this was true (at least in so far as it exists in the domain of the rule

base). If the user submitted the goal

?- line(X, b).

where the upper case X is a variable, this takes on the meaning

'What point X is on a line with b?'.

and requests PROLOG to find what point if any lies on a line with 'b'. PROLOG would match this goal against the database facts and instantiate - temporarily substitute - 'X' to 'a', and return the following

X = a

Yes

In a similar manner, if the submitted goal was

?- line(a,W).

PROLOG would answer

W = b

Yes

If both arguments are variables, PROLOG would return all those facts in the database with the functor 'line'. These examples demonstrate the non-deterministic and declarative nature of a PROLOG program and how it provides a flexible and powerful method of querying data in its database.

PROLOG also provides a highly modular approach to programming. Suppose the user wished to add the fact that

line(p,b).

This could be added to the database through a text editor. After re-compiling the

code, if the user then resubmitted the goal

?- line(X, b).

PROLOG would examine the entire database and respond

X = a

X = p

Yes

This would occur even if this new fact was added at the physical end of the existing database.

Rules are handled in exactly the same manner, i.e. by pattern matching terms against the database. For example, the user might want to know to what points 'b' was connected. The query would be

?- connect(X,b).

PROLOG would match this goal against the head (the clause before the ':-' symbol) of the 'connect' rule, and then 'call' the facts (or rules) in the tail (body) of the rule (the clauses after the ':-' symbol). PROLOG instantiates the arguments in the tail, one clause at a time, with the arguments submitted. Referring to Figure 4.1, it would first search the database for a fact which matched 'line (X, b)' and find that it matched 'line(a,b)', as it did above. The argument 'X' would then take on the value 'a'. PROLOG would return this solution to the user,

X = a

At this point, a goal has been satisfied. However, PROLOG is not necessarily finished. Because it searches the entire database and in this case has not done so, the

interpreter will attempt to find another solution to the goal 'connect(X,b)'. In this second round of goal seeking, PROLOG will find the recently added fact that 'line(p,b).' With X instantiated to 'p', PROLOG will return

X = p

PROLOG will continue to examine the database until every possible solution has been examined.

The 'connect' rules in the example program demonstrate recursion in PROLOG. The first connect rule in Figure 4.1,

```
connect(X,Z):- line(X,Z).
```

simply states that 'a point X connects to a point Z if X is on a line with Z'.

The second connect rule introduces recursion;

```
connect(X,Z):-
                    line(X,Y),
                    connect(Y,Z).
```

This can be interpreted as 'a point X connects to a point Z, if X is on a line with some point Y, and Y connects to some point Z'.

The goal

```
?- connect(a, What).
```

means 'What points are connected to point a' ? The manner in which this is resolved is to first call 'connect' rule 1, and then the tail of this rule, 'line(a,What)'. This goal succeeds against 'line(a,b)' and PROLOG will return

What = b.

to the user.

Since not all possibilities have been exhausted, PROLOG will try again to satisfy 'connect' rule 1. However, no other facts in the database match, and after all 'line' facts have been examined for 'connect' rule 1, PROLOG will give up on it. Since there is also a 'connect' rule 2, PROLOG will then attempt to satisfy it with the same arguments.

'Connect' rule 2 will first instantiate 'X' to 'a' and then call the first goal in the tail - 'line (a, Y)'. (The variable 'What' entered by the user is temporarily replaced by the variable 'Y' - in PROLOG, this is called the scope of a variable. Scope is limited to the current fact or rule. If and when the goal is satisfied, then the variable 'What' will be instantiated at the time of final solution). In the tail, PROLOG will find the fact 'line(a,b)' - as did rule 1 - and then instantiate 'Y' to 'b'. The next goal in the tail 'connect (Y,Z)' will then be called, with 'Y' set to 'b', and 'connect(b,Z)' thus becomes a goal. It is at this point that recursion begins. The previous clause in the database is a 'connect' goal, and the head and body of the current clause also contain a 'connect' goal; the clauses begin calling themselves. The result in this example is that 'connect' rule 1 is called, trying to satisfy 'connect (b,Z)'. In turn the tail of rule 1 is called, trying to satisfy 'line(b,Z)'; PROLOG finds in the rule base the fact 'line(b,c)'. 'Z' gets instantiated to 'c' and PROLOG begins to backtrack, i.e. move back along the path of goal seeking, and carry with it any instantiated variables to the calling clauses. It returns to 'connect' rule 2 with 'Z' set to 'c', and then to the first goal in the tail, then to the head of the clause, and finally instantiating 'What';

Who = c

At this point, PROLOG has determined relationships between points to two levels in the database; a is connected to b, and through b, to c. The next solution goes to a further level, and involves even more recursion. After the second level has been exhausted - when the recursive call to 'connect' rule 1 has failed to satisfy - PROLOG will call the second 'connect' rule, that is, it will call itself. In this example, after 'connect(b,Z)' fails in rule 1, rule 2 will be called with the same instantiation. PROLOG will then attempt to satisfy it in exactly the same way as it with 'a' as the first argument. That is, in the tail, 'line(b,Y)' is satisfied by 'line (b,c)'; Y is instantiated to 'c' and the second goal in the tail of rule 2 is instantiated to 'connect(c,Z)'. This calls rule 1 'connect(c,Z)' which is satisfied (in rule 1) by 'line(c,d)'. PROLOG begins to backtrack and the next solution is

What = d

This same procedure will occur until all solutions have been exhausted. PROLOG will continue to try all the other 'line' and 'connect' rules but will not be able to satisfy these, and will eventually fail and stop execution of the query.

This lengthy example of a simple PROLOG program serves to emphasize several important points about PROLOG and why it was chosen for the prototype. PROLOG is easy to modify, and as will be seen with UNCLE, the program or database is constantly being updated with new facts. To a certain degree the order of entry of rules and facts is not especially important. The programmer can enter simple facts at any stage - and at any physical location in the code. PROLOG

therefore provides a high degree of modularity in extending its database. However, some knowledge of the way in which PROLOG searches its database must be considered especially when recursive code is being used, since it is easy to produce cyclic recursion, the situation when two rules simply call each other in turn. It is relatively fast to develop and modify PROLOG programs, and PROLOG is therefore well suited for use in a prototype.

PROLOG provides a flexible method of querying a rule base. Users may have facts confirmed, or may request PROLOG to find solutions to a query in any number of ways.

PROLOG can model and determine relations very well. These may be spatial, as in the above example - or familial, cartographic, or otherwise. As will be seen in Chapter 6, UNCLE takes the commands of an ARC/INFO session, and models these as a lineage of events in a rule base. The lineage of any set of map features can be determined by applying similar concepts to the connectivity example given above, to GIS/cartographic objects and transformations. The relations are replaced by GIS or cartographic transformations, and the arguments become sets of map features. By defining these relations, extending these to include uncertainty measures, and then applying recursive processing, UNCLE is able to carry these measures through the many transformations of GIS software.

PROLOG has been used to implement uncertainty measures in other rule-based systems. Hinde (1986) describes the use of PROLOG to model fuzzy measures of uncertainty; Baldwin (1986) has developed a Fuzzy Relational Inference Language

(FRIL), an extension of standard PROLOG which incorporates fuzzy logic. Bratko (1990) describes the use of probabilistic or subjective uncertainty measures (certainty factors) in expert systems applications. Zhang (1989) has applied PROLOG to problems of maintaining consistency, a type 4 uncertainty, in cadastral data.

In PROLOG the declarative nature and powerful recursive capabilities of the language greatly expedite the modelling of complex relationships such as those found in GIS databases.

PROLOG meets Requirements 5 and 6, and to a lesser degree, Requirements 1-4. For all of the above reasons, it has been adopted for the prototype.

The PROLOG software chosen for this research is BIMProlog (BIM, 1990). It is available at relatively low cost to educational institutions, and can run on the same hardware platform as ARC/INFO.

4.2.3 Other Languages

Prolog has some ability to perform numeric computations, character manipulations, and file input/output, and is used to a limited extent in the prototype for these. However, these are not especially efficient or easy to use in Prolog. Typically these types of tasks are very straightforward and procedural in nature, and other languages are more suitable for them. FORTRAN and C are used for the development of many of the translators for this research, which have involved a great deal of file input and output, interpretation of string data, and reformatting of numeric data. These other languages are also better for representing large sets of

numbers in regular arrays and in handling more complex mathematics of uncertainty, such as statistical analysis. It is therefore important that these languages be accommodated in the prototype, to meet Requirement 1, for numeric computation. The programming for uncertainty in this research has been done in PROLOG, however, there is some investigation of how to utilize these other languages in conjunction with this task.

4.2.4 Operating System Software and Utilities

The UNIX operating system has been in widespread use in academic institutions for a number of years. Recently it has begun to spread rapidly into industry and government. With the rapid growth of the workstation market, with which UNIX is associated, and the move by database, GIS and other software vendors to this type of hardware and software platform, UNIX appears to be the operating system of choice for the engineering marketplace, and especially for GIS, in the next decade.

UNIX is a true multi-tasking operating system. Multi-tasking refers to the ability of the system to carry out a number of processes, or tasks, at the same time. In terms of the prototype, this is important. Neither ARC/INFO, PROLOG or another language meet all of the requirements identified above. Used together, however, they may provide all of the required functionality. In UNCLE, the approach is to use the multi-tasking capability of UNIX, by running concurrent processes for these different software.

For the end user utilizing UNCLE, these separate processes are accessible through a windows-based graphical user interface (GUI). UNIX allows for multiple windows to appear on the computer monitor, with a different process running in each one. UNCLE most frequently will display two or three windows with ARC/INFO running, and two with BIMprolog running. Within either of these, additional processes may be occasionally, and temporarily, executed.

The version of UNIX used is SunOS, a hybrid System V/BSD UNIX developed by SUN Microsystems and implemented on their line of SPARCstations. The GUI used is SUNVIEW, which is provided with the operating system. Both ARC/INFO and BIMprolog work under SUNVIEW.

UNIX, like most operating systems, provides the capability to group operating system commands into files, and execute these as a program. Under UNIX, these are called scripts, and are analogous to MS-DOS batch files or VAX VMS COM files. However, the programming capability of UNIX scripts is much greater than either of these. The power of scripts can be further enhanced by the use of a number of system utilities provided with UNIX. Such programs as AWK, GREP and TR, for example, can (respectively) help manage file data in fixed field formats, find strings within files, and translate characters in files. In addition, UNIX is capable of interprocess communication. For example, the output of one utility, program or script can be 'piped' into another as input. The advantage of these tools is that they provide very quick and relatively easy manipulation of ASCII data stored in files, and can in many instances do away with the need to develop a C or FORTRAN program.

Consequently, they are of great use in prototyping, and have been used to develop UNCLE.

4.2.5 Procedures and Rule Base in UNCLE

The individual programs and processes outlined above communicate to one another through a number of intermediate data files. A number of procedures have been developed and implemented using AMLs, operating systems scripts, and programs in C, to move data from one part of UNCLE to another. For example, an ARC/INFO AML calls a UNIX script which "moves" the data from an ARC/INFO file format into a format which is readable by the PROLOG rule base. The user can then move into the BIMprolog process window, add the rules and then query the rule base on such things as lineage of a coverage, accuracy of source documents of a coverage, etc..

4.3 Summary

The computing requirements for a prototype system to explore uncertainty in GIS are identified. A configuration is described, consisting of a suite of GIS software, UNIX operating system commands and utilities, PROLOG and other high level languages, and procedures for UNCLE. These have been chosen to satisfy the different requirements for investigating the four types of uncertainty previously described. PROLOG plays an extremely important role in UNCLE, and the most relevant features of it are described in greater detail.

Chapter 5

Database Development

5.1 Introduction

This chapter describes the development of the GIS database used in the research. The development followed these steps:

a) selection of a study area,

b) determination of data and data sources,

c) data preparation:

i) acquisition (digitizing),

ii) data validation (cartographic data),

iii) data validation (attribute data).

5.2 Study Area

The area selected for this study lies in the Kananaskis Valley, located in the Rocky Mountains of Alberta, approximately 130 kilometres west of the City of Calgary, and bordering on the province of British Columbia (Figure 5.1). It is a magnificent area of steep mountains and flat valleys, with significant recreational, wildlife, and environmental importance. Excellent detailed descriptions of the physical geography of the valley are available in Paine (1983) and Mulaku (1987).

The portion of the valley utilized is the same as that used in Mulaku (1987).

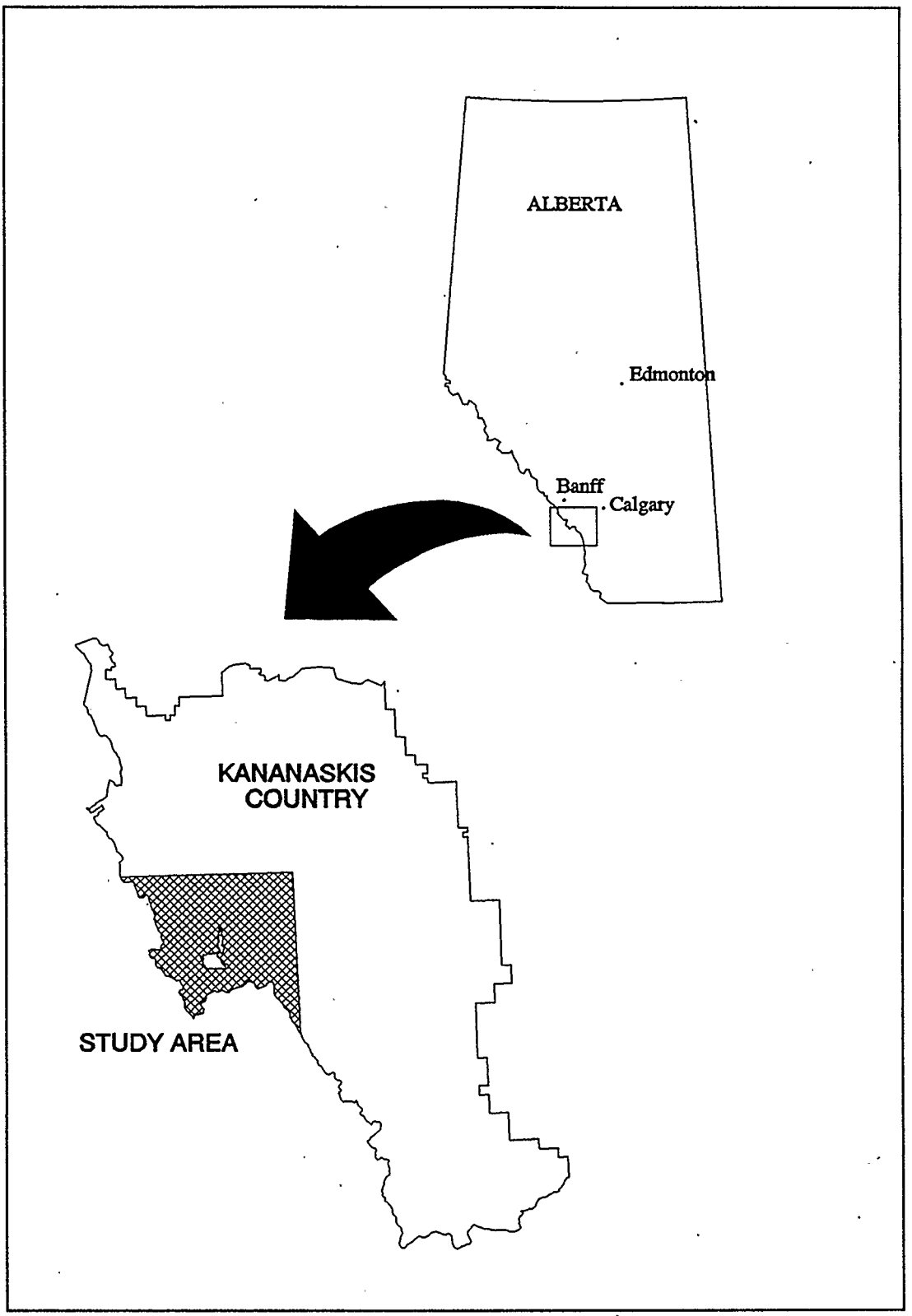

Figure 5.1 Location of study area

This sub-area is approximately 30 km by 30 km, and lies between latitudes 50° 20' N and 50° 50' N, and longitudes 114° 50' W and 115° 50' W, on the Alberta side of the Alberta-British Columbia provincial border.

The Kananaskis valley is the site for several educational activities for The University of Calgary; the Department of Surveying Engineering holds their annual summer field camp for undergraduates there, and frequently conducts research projects in the area. The valley was selected primarily because of previous Departmental research in GIS-related topics in the area. Paine (1983;1987) has investigated the use of digital Landsat data for surface cover mapping, and for information extraction and integration into land-related information systems. The valley is also the site for proposed development of a land-related information system based on Landsat data (Lodwick et al, 1986). Mulaku (1987) researched problems associated with map data digitizing and developed algorithms for automated hydrological network reconstruction.

This thesis contributes to teaching and research efforts in a number of ways. First it develops a database for use with GIS software. This consists of spatial and attribute data of some of the natural and cultural features of the valley. This is provided in a generic data format, as well as in formats for three different GIS packages. These can be used for undergraduate teaching, and also for future research. Second, it documents some of the data structures required for different GIS software packages, and develops a number of utility programs for exchanging data between them. Third, it examines additional aspects of issues raised in previous work,

such as solving problems of sliver polygons (Mulaku 1987), and topological consistency in data (Mepham 1988; Zhang 1989). Last, and most importantly, it develops a prototype machine for investigating uncertainty, using a rule-based coupled systems approach; this is the subject of Chapter 6. Associated with these has been the development of a number of tools, in the form of algorithms and procedures, which enhance the utility of existing Departmental facilities, either for the purposes of extending the database, or for furthering this or related research.

5.3 Determination of Data Required and Data Sources

Figure 5.2 depicts the general procedures followed to acquire digital map data for entry into the GIS. These steps are described more fully in the following sections.

In previous research, maps of soils, bedrock geology, surface geology, cadastral and hydrographic features in the study area were digitized (Mulaku 1987). These were examined for possible adoption as the data set. Although the hydrographic data set, consisting of lineal and polygonal features, proved complete in geographic extent and topologically consistency, an examination of the remaining available polygonal data confirmed two problems. First, the cadastral data set was incomplete, due to the lack of Range/Township maps available to the researcher (Mulaku, 1987). The second problem was that the polygon data for soils, surficial geology and bedrock geology contained duplicate line work. This was a result of digitizing each polygon individually, without regard to common polygon boundaries. Consequently, the soils and geology features were re-digitized, with single line representation for polygon

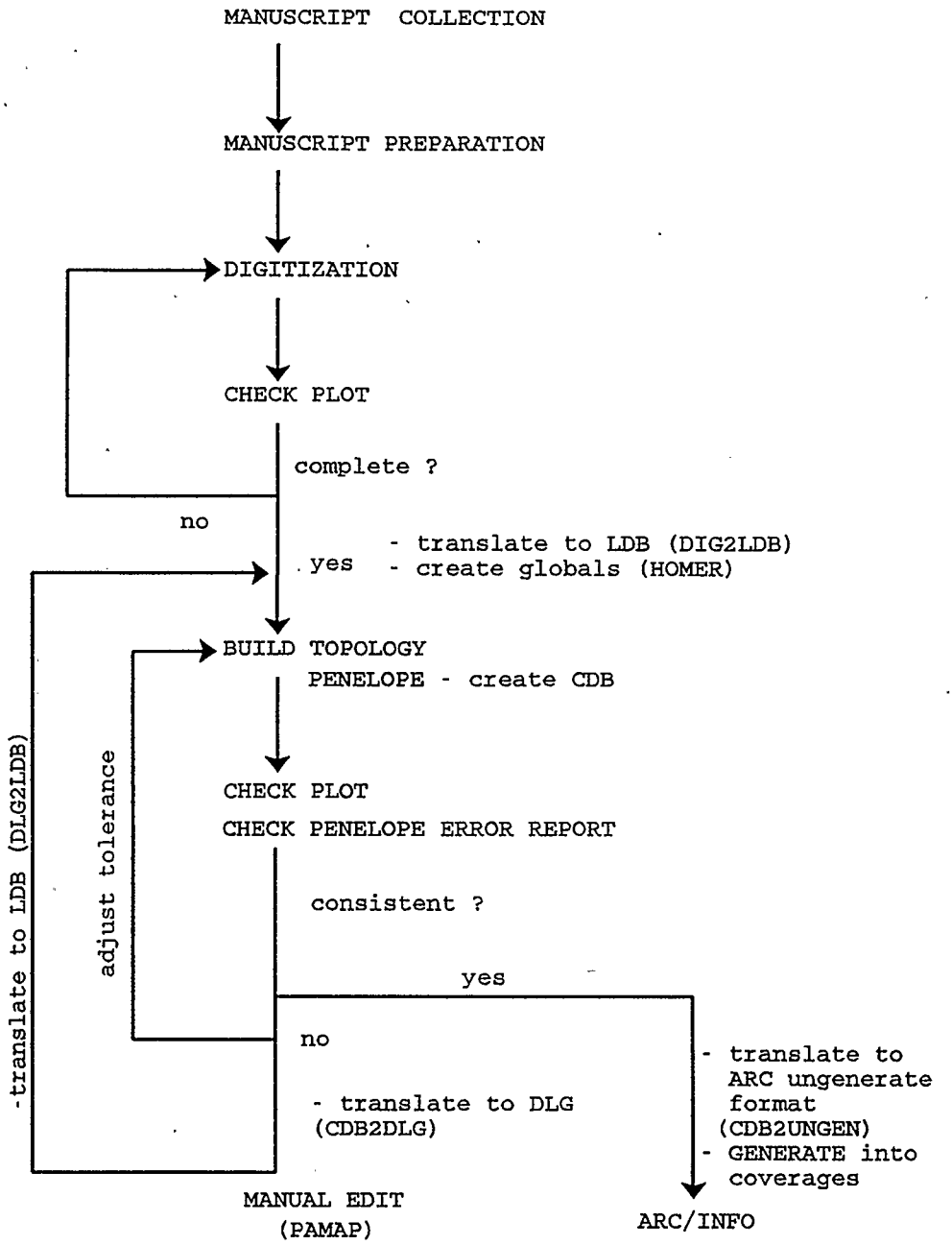

Figure 5.2 Stages in preparation of data for entry into GIS

boundaries. The hydrography and cadastral data were adopted as is, and processed along with the new and re-digitized data. In addition, it was decided to digitize an additional set of cultural features, the boundaries of the integrated resource planning areas of the park.

The following two paragraphs describe the sources for the hydrographic data and cadastral data, respectively.

"NTS maps 82J/6,7,10,11,14 and 15, published by the Canadian Department of Energy, Mines and Natural Resources at an original compilation scale of 1:50,000. Also acquired was NTS map 82J by the same publisher at the derived scale of 1:250,000..." (Mulaku, 1987, p. 85).

"Cadastral maps covering Township 19/Ranges 7 and 8, Township 20/Ranges 8 and 9 and Township 21/Range 9. These are published by the Government of Alberta, Department of Highways at a compilation scale of 40 chains to an inch (about 1:32,000)..." (Mulaku, 1987, p. 86).

The administrative, soils and geology data were available only in analog map form. These were borrowed from The University of Calgary Main Library, Map and Airphoto Section and digitized as described in section 5.4. Table 5.1 summarizes the features and their associated map sheet, scale and compilation source.

Feature	Map Sheet	Scale	Source
Soils	82J - Kananaskis Lakes, Canada Land Inventory Soil Capability for Agriculture	1:250000	Department of the Environment
Surficial geology	Sheet 5 Surficial Geology, Alberta Foothills and Rocky Mountains	1:250000	Department of Environment and Natural Resources, Government of Alberta
Bedrock Geology	Sheet 5 (inset) Surficial Geology, Alberta Foothills and Rocky Mountains	1:1000000	Department of Environment and Natural Resources, Government of Alberta
Integrated Resource Planning Areas	Kananaskis Country Integrated Resource Plan	1:100000	Alberta Energy and Natural Resources, Resource Evaluation and Planning Division
Hydrographic	NTS map sheets 82J/ 6,7,10, 11,14,15	1:50000; 1:250000	Department of Energy, Mines and Natural Resources
Cadastral	Township 19, Ranges 7,8; Township 20, Ranges 8,9; Township 21, Range 9	1:32000	Department of Highways, Government of Alberta

Table 5.1 Map features prepared for study area.

5.4 Data Preparation

5.4.1 Map Digitizing

Maps were manually digitized on a Summagraphics Corporation digitizing tablet connected to a VAX 11/750 minicomputer. The digitizing software used was the Department's DIGIT program. During digitizing, data were displayed on a

Tektronix 4208 graphic display.

The map registration procedure consisted of physically mounting the map on the active digitizing area of the tablet, and invoking the DIGIT program. DIGIT establishes digitizer to map coordinate system transformation parameters using a two-dimensional affine transformation. This requires the user to enter the known coordinates of four to ten control points on the map manuscript and then enter the positions of these points on the tablet using a cursor to provide the coordinates. DIGIT then performs the transformation and produces a summary report which includes the transformation parameters (scale factors in X and Y, rotation, translations and the non-perpendicularity (skewness) of the map axes). It also includes an RMS error of the registration procedure.

These map registration data are incorporated into an ASCII file which DIGIT produces during each digitizing session. Additional data can include the time and date the file was opened, coordinates digitized, and user-entered codes for the digitized data. An example of a DIGIT file is shown in Figure 5.3. These files can be edited by the user, thus allowing for additional comments to be entered after the session is complete, and allowing easy access to coordinate data.

Two features of DIGIT files are important to this research. First, the map registration information in these file forms the basis for 'history' files which are utilized later in the prototype. Second, the digitized coordinates in these files are extracted for data validation and processing in other programs.

The digitizing process was done in a "spaghetti" fashion, i.e. no topological

```
!     FILE OPENED ON  6-MAR-89 AT 17:29:08.
!
! ***** MAP MOUNTING INFORMATION *****
!
!    MAP NAME: kv_irpa              MAP UNITS: METRES
!
!  Point ID    Northing       Easting       X-Dig      Y-dig      Res-X     Res-Y
!   LL        5595081.230    606382.806      819.6       52.7     -0.69      0.44
!   UL        5650677.120    605255.425      262.8       55.8      0.69     -0.44
!   UR        5624786.212    676359.770      538.4      756.7     -0.70      0.45
!   LR        5569199.033    678232.493     1092.5      762.8      0.69     -0.44
!
!                                                        RMS       0.69      0.44
!                                                  TOTAL RMS            0.58
!
!     SCALE FACTORS      NORTH  100392.814     EAST  100117.577
!     OFFSETS            NORTH 5676927.343     EAST  599059.059
!     ROTATION                -88.74 Deg.
!     NON-PERPENDICULARITY     0.13 Deg.
!
!
!  Seq. Num.    Northing     Easting   Elevation  Code  Pen  Annotation
          1 5658424.115  612362.407       0.000  UNKNWN  1
          2 5658449.525  613968.545       0.000  UNKNWN  1
          3 5656738.015  614010.105       0.000  UNKNWN  1
          4 5656783.443  615615.756       0.000  UNKNWN  1
          5 5655152.667  615685.483       0.000  UNKNWN  1
          6 5655181.176  617432.136       0.000  UNKNWN  1
          7 5653599.560  617460.500       0.000  UNKNWN  1
```

Figure 5.3 Text file produced by DIGIT, showing registration data and digitized coordinates. The ! character denotes a comment.

data (polygon identifiers, left/right relations, connectivity of linework) were entered as the cartographic linework was being entered. Each linestring which formed a polygon boundary was digitized once. Also, no attribute data for polygons were entered.

5.4.2 Data Validation (cartographic data)

Analog to digital data conversion and validation are typically lengthy and expensive stages in the development of a GIS database. They are also obviously critical stages, in that later use of the data in the GIS requires a complete and consistent set of data. Blunders in the data acquisition stage will seriously affect the

integrity of the database and consequently, the accuracy of results.

The first stage of the data validation process was to create a topologically correct set of polygons from the digitized map data. It involved an iterative process, and required checking the completeness of the digitized data, reformatting of data for importing to other programs, processing for polygon creation and topology, the removal of minor digitizing errors, visual proofing of results, manual correction of some errors, and then reprocessing. Figure 5.2 includes details on the steps and tools used to achieve this.

Visual checking of digitized linework

The first step was to check that all linework had been digitized. This was accomplished by re-plotting the raw digitized data at the scale of the original map manuscript on translucent paper for comparison with the original. The paper was then placed on the original map and common features were used to register the plot with the map. This produced a graphical overlay. The linework was then compared. If all lines were verified as present in the proof plot, the digital data was then passed on for topological processing; if not further digitizing was required until all lines had been captured.

Topological Processing using ODYSSEY

The ODYSSEY GIS (ODYSSEY, 1982) was used for topological processing of the data. It was used because it was the only GIS software with this capability available to the author during the database development stage. ODYSSEY is a collection of computer programs for the entry, analysis and display of geographic

data. Developed at the Harvard Laboratory for Computer Graphics and Spatial Analysis, ODYSSEY represents some fifteen years effort in program design and implementation. The result is a set of highly integrated program modules which can manipulate point, line and area data in a wide variety of operations. These operations include digitizing from existing map manuscripts, data manipulation (eg. line generalization), data analysis (eg. point in polygon and polygon overlay) and drawing of shaded-area, perspective view and line maps.

ODYSSEY incorporates the results of some of the seminal research work in computer cartography of the 1970s, for example the topological chain structure developed by Peucker and Chrisman (1975), and the polygon overlay processing algorithms documented by White (1978).

ODYSSEY procedures and modules were employed as tools in the data validation process. For example, it is relatively easy in ODYSSEY to import and export "foreign" cartographic data. Once data are in ODYSSEY format, the PENELOPE module can generate topologically sound chain files, with topological encoding for polygons, from digitized spaghetti line work. The open architecture and accessibility to ODYSSEY source code made possible the addition of a customized device driver for the Department's Datatech 3454 pen plotter, for visual proofing of intermediate results.

In order to utilize ODYSSEY for creating polygon topology on the digitized map data, it was necessary to bring the data into ODYSSEY file structures. Cartographic files in ODYSSEY are considered to be in two parts. The first part is

the data part, which contains the actual cartographic data, i.e. the spatial component such as chains (line strings). These are stored with the extension "LDB" or "CDB", respectively. Figure 5.4 shows the content and structure of a chain file. Other cartographic data files in ODYSSEY are the values and cross reference files which contain attribute or spatial relations information (eg. polygon subsets). Identifiers of the cartographic entities in both types of file act as indices. For example, the data for polygons will appear in polygon data files, which contain, for every polygon, a list of the chain identifiers which bound the polygon. When an operation such as a computation of a polygon area is to be performed, the list of chains is read, and the chain identifiers are used to access the chain records in the CDB files.

Data files can be written in either binary or formatted (ASCII) format; the former is typically used for the geometric data to reduce disk storage requirements and the latter for the attribute files. Additional formats for the different data files are included with the ODYSSEY GIS documentation (ODYSSEY, 1982) and are straightforward and consistent in design.

The second part of cartographic files in ODYSSEY are globals files. These are essentially meta-data files and contain information about the LDBs or CDBs, values files, etc.. This includes the type of file, file name, blocksize, the format fields, etc. ODYSSEY accesses LDB (spaghetti line data) and CDB (topologically connected lines) files from the globals. All of the ODYSSEY modules first read the globals files and use the information in them to read the associated data or value/cross reference file. Globals files for coordinates are stored with the extension

a) Example of an ASCII CDB file.

```
        1       20        19        20         0         6
 6.381650E+05 5.604223E+06 6.379660E+05 5.604117E+06 6.378440E+05 5.603987E+06
 6.378000E+05 5.603759E+06 6.375660E+05 5.604030E+06 6.374060E+05 5.604353E+06
 6.372210E+05 5.604676E+06 6.370620E+05 5.604949E+06 6.369550E+05 5.605173E+06
 6.366760E+05 5.605241E+06 6.364240E+05 5.605259E+06 6.361990E+05 5.605203E+06
 6.359750E+05 5.605096E+06 6.358530E+05 5.604966E+06 6.356560E+05 5.604809E+06
 6.355340E+05 5.604680E+06 6.352610E+05 5.604546E+06 6.350360E+05 5.604489E+06
 6.349120E+05 5.604410E+06 6.346360E+05 5.604377E+06
        2       42        20        50         9         6
 6.346360E+05 5.604377E+06 6.345280E+05 5.604652E+06 6.343710E+05 5.604849E+06
 6.342660E+05 5.605023E+06 6.340840E+05 5.605220E+06 6.339550E+05 5.605342E+06
 6.338250E+05 5.605465E+06 6.337690E+05 5.605665E+06 6.337380E+05 5.605891E+06
 .......

-999999999        -1-999999999-999999999-999999999-999999999
```

b) Each chain record consists of a header record and the coordinate list for the chain.

	Bytes	Field
Header record	1-10	Chain identifier
	11-20	Number of coordinates
	21-30	Start node identifier
	31-40	End node identifier
	41-50	Polygon left identifier
	51-60	Polygon right identifier

Coordinates are stored as a continuous list of x,y values, 13 bytes for each coordinate value (FORTRAN format F13.6).
End of file is indicated by unique trailer records.

Figure 5.4 Content and structure of an ASCII chain file produced by ODYSSEY

CDG (for "Coordinate Data Globals").

External files of cartographic data can be imported into an ODYSSEY LDB format by first reformatting the data to an ASCII file whose record structure is ODYSSEY-compatible for that entity type. This is done for the sample data with the program DIG2LDB. The HOMER module is then used to generate the globals file for the line file. This is achieved by using an existing, generic globals file as a template. The generic globals file is modified to include the filename of the line file to be converted, that is the file produced by DIG2LDB. An ODYSSEY command

sequence is then given which utilizes this modified globals file to read in the reformatted file and copy it out to disk again. In this process the globals file is updated and a proper LDB are created. After this is done, any ODYSSEY module can access the data.

To create polygon topology from the digitized line data the PENELOPE module was used. PENELOPE will automatically generate from a line file and its globals file (the LDG/LDB pair) a topologically correct chain file (CDG/CDB), with automatic polygon numbering. Tolerance distances for automated elimination of overshoots and undershoots can be set during this process, to replace the default tolerance, which is initially set to 1/1000 of the map extents. PENELOPE produces an error report for each polygon file it produces. In conjunction with proof plots, these reports were used to find topological errors (see Figure 5.5), and guide further processing.

The designers of ODYSSEY anticipated the need to interface the program to different hardware configurations. Consequently, they have included well documented source code and examples for interfacing to unsupported devices. Although writing device drivers for ODYSSEY requires some knowledge of FORTRAN programming and the specific device hardware, the actual modification or addition of the code is straightforward. For data validation, the visual proofing of data required a hard copy plot. The Departmental plotter, a Datatech 3454, is not supported by ODYSSEY; consequently, a device driver was written for it. This was incorporated into the source code of programs DRWING.FOR and DEVICE.FOR, and the complete ODYSSEY

```
PENELOPE ERROR REPORT                                          2APR89

LINE CONVERSION

 INPUT FILENAME AND TITLE:
   SOIL_1.LDG
   STANDARD GLOBALS FILE FOR ODYSSEY FILE

 OUTPUT FILENAME AND TITLE:
   DUA0:[SCRATCH]SOIL_1.CDG


OUTPUT POLYGON IDENTIFIERS       NEW
TOLERANCE DISTANCE      7.500000E+01     TOLERANCE PERCENT      0.177242

CHAINS WITH IDENTICAL LEFT/RIGHT POLYGON IDENTIFIERS
      CHAIN ID      COUNT  FROM NODE    TO NODE POLYGON ID        LENGTH
            16          3        134        132           6  2.294040E+02
```

Figure 5.5 PENELOPE error report for soils data, showing the conversion of LDG/LDB "spaghetti" to CDG/CDB chain topology. Tolerance values are reported, and chains and polygons can be identified by number. Distances in metres.

modules were recompiled. This makes the device 'DT3454' transparently accessible from the ODYSSEY modules. The PENELOPE-processed polygons were plotted with the polygon and chain identifiers to permit easy identification of topological inconsistencies.

Although many small digitizer errors are corrected automatically it is nevertheless necessary to monitor the process of data validation carefully. The main reason for this is that the improper use of tolerance values can produce unwanted results, as nodes or vertices of linework which fall within the active tolerance value of each other are automatically coalesced.

The typical approach to building topology on a data set is to start with a small tolerance. The results are then examined. If there are significant large errors remaining, it may be appropriate to increase the tolerance value and reprocess.

However, there may also be good reason to perform isolated editing on selected areas to prevent unwanted coalescence of points, during this iterative process. It is desirable to be able to interactively edit in these situations, using CAD techniques for zooming in and out, graphical selection and deletion of lines, etc.

Unfortunately, neither the DIGIT program nor ODYSSEY provides a suitable environment for editing cartographic data. DIGIT is used solely for data input and ODYSSEY is inflexible for graphics and editing. The ARC/INFO GIS software, and its powerful interactive editing environment ARCEDIT, was not available for use at this stage of the research. Consequently, another GIS software package, PAMAP, was employed to provide the interactive editing capability.

PAMAP GIS was originally designed to facilitate the updating of forest inventory maps. It is available on the Departmental VAX 11/750 computer, and so was easy to use in conjunction with the other programs. PAMAP provides vector-based entry and interactive editing of data, and raster analysis. It also includes a number of translation capabilities for data formats; one of these allows for the import and export of Digital Line Graph (DLG) files. DLG is the main data format for distribution of U.S. Geological Survey digital data, and is commonly used in North America.

In order to make use of PAMAP for topological checking and editing, it was first necessary to convert the CDB files produced by PENELOPE to DLG format. Program CDB2DLG was written for this purpose. Data was successfully converted and brought into PAMAP, edited where necessary, and exported back to DLG. It was

brought back to ODYSSEY LDB format with another translator program, DLG2LDB. The data set was then resubmitted to PENELOPE for verification, as outlined above.

This validation process was repeated for each of the four digitized maps, until a complete, topologically consistent data set was established. The data were then archived to 9 track tape, and transferred to the UNIX based SUN network at the British Columbia Institute of Technology in Burnaby, B.C..

A FORTRAN program, CDB2UNGEN was written to translate the CDB files from ODYSSEY chains to ARC/INFO UNGENERATE format, which are ASCII files of line strings. In ARC/INFO, the term 'arc' is used to denote a single line segment or a sequence of line segments joined end to end, and is therefore the same as an ODYSSEY chain or line string. The 'ungenerate' files were converted to arcs in ARC/INFO coverages using the GENERATE command with the LINE option. Separate coverages were created for each set of data. Since the ungenerate format does not support the encoding of topological relationships, the topology was lost during the conversion process, and it was necessary to rebuild it. In order to build polygon topology, it is necessary to label each polygon. This was performed satisfactorily in ARCEDIT, and the coverage topology was rebuilt using the CLEAN command with the POLY option and a tolerance of 0.001 metres. At this tolerance, all polygons were rebuilt as they were in the original ODYSSEY derived CDB files.

5.4.3 Data validation (attribute data)

The attribute data for all of the different polygons digitized were loaded into the INFO database by manual methods. An INFO table was defined with two fields for each polygon. The first field held the polygon identifier (derived from ARC/INFO), and the second field contained the attribute data (taken from the manuscript). The polygons were plotted with the attribute, and once again a visual proofing was performed to verify the data.

5.5 Summary of GIS Database

At the conclusion of the database development process, seven coverages had been created in ARC/INFO. Table 5.2 summarizes the ARC/INFO coverages and their contents. One of these, KVLAKES, was derived directly from the HYDRO coverage, and contains only Upper and Lower Kananaskis Lakes.

The coverages exhibit many different kinds of uncertainty. They are derived from different source documents, of different scales, with different levels of positional accuracy. Except for the cadastral data set, which was known to be incomplete in previous research, all are complete. Some of the polygon data are categorical or discrete in nature, such as the cadastral and administrative boundary coverages. Others are more continuous, such as soils and geology. The re-digitized coverages have associated 'hist' files, based on the DIGIT files, which describe how the coverage was created, and contain meta-data about the coverages. These are described in Chapter 6.

COVERAGE NAME	CONTENTS	FEATURE TYPE	TOPOLOGY	ATTRIBUTES
HYDRO	Hydrological features (lakes, streams, rivers)	Arcs; polygons	Arc	No
CADASTRAL	Parcels	Arcs; polygons	Arc	No
SGEO5	Surface Geology	Polygons	Polygon	Yes
BGEO5	Bedrock geology	Polygons	Polygon	Yes
SOILS82J	Soils	Polygons	Polygon	Yes
KVADMIN	Integrated Resource Planning Areas	Polygons	Polygon	Yes
KVLAKES	Upper and Lower Kananaskis Lakes	Polygons	Polygon	Yes

Table 5.2 Summary of ARC/INFO coverages.

5.6 Summary

The Kananaskis Valley of western Alberta was chosen for the study. Previous research in the area provided hydrological and cadastral data. Additional geology, soils and administrative data were manually digitized from map manuscripts. The validation procedure involved a careful iterative process of automatic and manual editing. The final result was a complete and consistent set of polygons for four different maps. All data were transferred to ARC/INFO GIS and the topology verified with that software. Attribute data for each polygon were extracted from the maps and added to the INFO database management system.

Chapter 6

Experimentation

6.1 Lineage in Geographic Information Systems

Most studies of uncertainty in GIS have focused on specialized aspects of the problem, such as the accuracy of manual digitizing, or error in polygon overlay. However, it is also important to be able to represent the uncertainties of these transformations in a computer and be able to track these through several geoprocessing operations, utilizing data on source and lineage (Miller et al, 1989; Lanter and Veregin 1990). Comparatively little research has been done in this area. In keeping with the communication paradigm of GIS proposed by Bédard (1987) and adopted in Chapter Two, uncertainty should be studied as a sequence of transformational models from source to user. To achieve this, this research takes the approach of studying the lineage of data in a GIS.

Lineage includes data about the original source material and all the processes and transformations leading to the final digital data base product (NCDCDS, 1988). Lineage provides meta-data (data about the GIS data) and is an example of a type 4 uncertainty (section 2.3).

The issue of data lineage is critical in order to ascertain the validity and suitability of GIS data for a particular use (Grady, 1988). Nevertheless it is one that is rarely included in software in any sort of automated fashion.

The main focus of UNCLE is to model lineage in a GIS. This facilitates

investigation into modelling other kinds of uncertainty, and error propagation. UNCLE provides techniques to incorporate and partially automate lineage reports in a GIS environment. These reports are based on data derived directly from the ARC/INFO GIS data sets from the study area, and is accomplished by extending the ARC/INFO GIS model to include a PROLOG rule-base. This rule base provides for the addition of source material data and definition of relationships between data sets. This establishes a meta-level treatment of data, which is lacking in conventional GIS. This chapter describes the development of the computing environment of UNCLE.

6.2 ARC/INFO Architecture and Data Organization

The ARC/INFO GIS is a set of general purpose software modules for managing geographic data. It was designed as a "toolkit" of geographic operators, and provides users with a large degree of flexibility in customizing the software for more specific purposes. ARC/INFO organizes geographic data into "coverages", and users manipulate these from a "workspace". Coverages typically represent a set of phenomenologically related geographic features in a given area, for example, streams, lakes etc.. Coverages include spatial and attribute data about the features. These are stored, and usually manipulated, as a set. A workspace is a directory in the computer file system which contains one or more coverages. Within a workspace, the various coverages will usually span the same geographic area.

Due to the large geographic extent and size of many GIS databases, it is common to partition the data into geographic areas and provide the user with data

management tools for these. This may be done for reasons of efficiency, or to model an existing system of mapping. In ARC/INFO these geographic areas are called "tiles" and the MAP LIBRARIAN provides such management capabilities. When such a partitioning scheme is employed, ARC/INFO coverages are further organized into user-defined tiles; the collection of tiles is called a "layer". In this thesis, the study area is not partitioned, and so a coverage and layer mean essentially the same thing; consequently LIBRARIAN is not needed. In keeping with ARC/INFO terminology, 'coverage' will be used to describe sets of features.

Workspaces and coverages are implemented in the computer operating system as directories and sub-directories in a hierarchical file system. The workspace directory contains a number of sub-directories, which are the coverages, and an 'info' directory. Figure 6.1 gives an example of a basic organization of a workspace and its coverages. The user will operate the software from the workspace level, or at a higher level in the file system hierarchy, with a file pointer set in ARC/INFO to the appropriate workspace directory.

In addition to the coverage and INFO sub-directories, a workspace may contain a number of other files. These may be ARC/INFO files such as arc macro language (AML) programs or records of database transactions stored in LOG files. User files may also be stored here.

Before describing the file organization at the coverage level, it is helpful to understand how spatial and non-spatial data are organized and managed in ARC/INFO. Like many other GIS software packages, ARC/INFO utilizes a

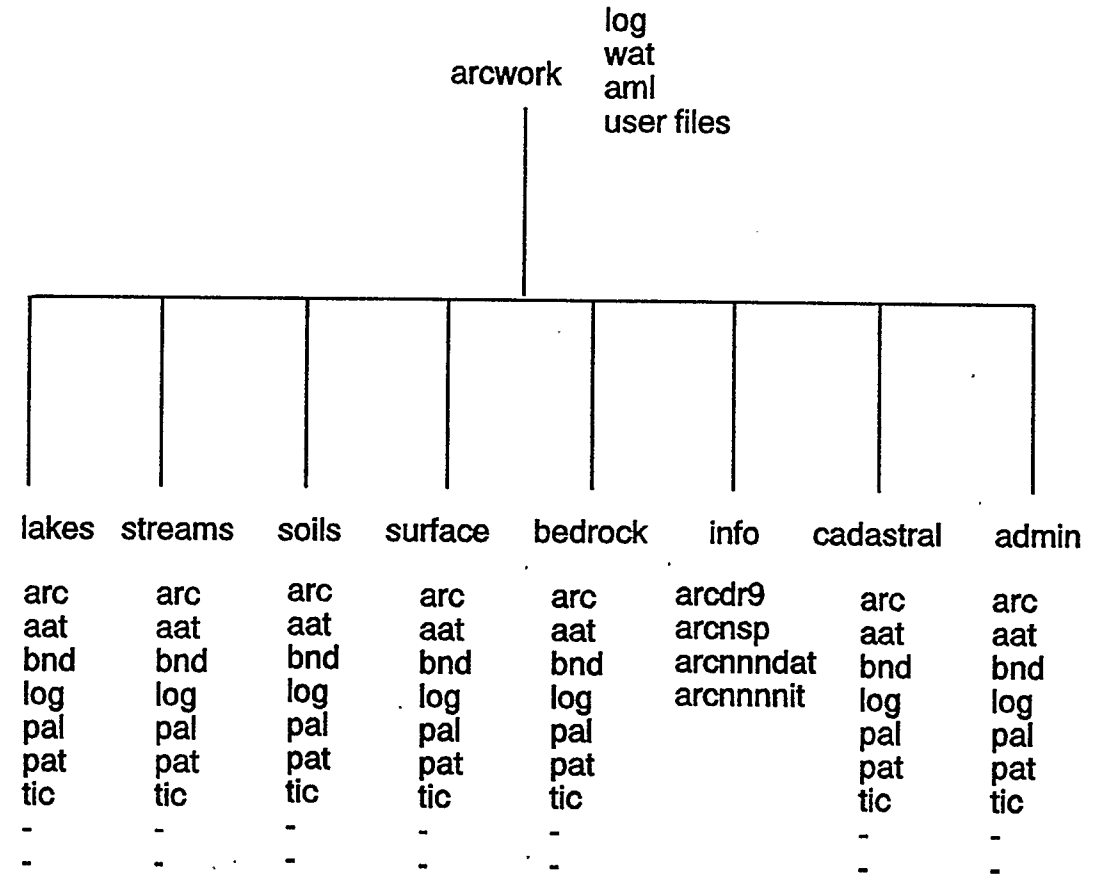

Figure 6.1 Organization of an ARC/INFO workspace, showing the workspace directory ('arcwork') and the coverage and 'info' directories and some common ARC/INFO files.

relational database management system in conjunction with a non-standard, internal file management system. In ARC/INFO the relational database management system (RDBMS) is INFO, produced by Henco Software, and it manages data in a number of database tables or relations. Often the term "non-spatial" is applied to the data in

the RDBMS; however, this is not strictly true in the ARC/INFO case. The tables in INFO will certainly include, if necessary, non-spatial attribute data supplied by the user. However, there are also a number of special tables used in conjunction with the internal file system; these are managed entirely by the ARC/INFO software, and contain a reference to the "true" spatial data in the internal file system. They may also contain spatial data themselves.

In the internal file management system, ARC/INFO stores and manages the spatial definitions for the features. This includes the coordinate lists and topology for "arcs" (linestrings) and the detailed definition of polygon topology. Also included are a number of cross-reference files. Internal files are direct access binary files and their file structure is confidential and proprietary, and so is not available in the standard documentation.

GIS software such as ARC/INFO is able to manage and manipulate internal spatial data and associated RDBMS data, by providing and maintaining a constant link between them. In ARC/INFO this link is an internal feature identifier (an integer value) in an INFO table, and the corresponding feature identifier value stored in the internal spatial files. Any time a feature's definition is changed its associated records throughout the set of internal and INFO files require updating.

At the coverage level of data organization, INFO tables are stored in a sub-directory of the workspace called 'info'. The 'info' sub-directory maintains a number of files (see Figure 6.1). The most important of these are;

'arcdr9', which contains a list (directory) of all of the INFO files for the workspace,

'arcnsp', which specifies the current output device, for example, the computer monitor or a disk file,

'arc*nnn*dat' files, which contain the names of data files, as used internally by INFO, and

'arc*nnn*nit' files, which contain the names of items (fields) in the INFO files.

In the latter two types of files the '*nnn*' represents a three digit code assigned by INFO. These filenames are cross-referenced in the 'arcdr9' file to the filenames which the user enters to access data. Figure 6.2 gives an example of the information contained in an INFO 'arcdr9' file and the arc*nnn*dat files and their corresponding user names. The INFO directory may also contain external INFO tables, created by the user, and maintained by INFO.

TYPE	NAME	INTERNAL NAME	NO. RECS	LENGTH	EXTERNL
DF	BGEO5.TIC	ARC002DAT	4	12	XX
DF	BGEO5.BND	ARC003DAT	1	16	XX
DF	KVLAKES.TIC	ARC004DAT	4	12	XX
DF	KVLAKES.BND	ARC005DAT	1	16	XX
DF	KVADMIN.TIC	ARC006DAT	4	12	XX
DF	KVADMIN.BND	ARC007DAT	1	16	XX
DF	BGEO5.AAT	ARC008DAT	28	28	XX
DF	HYDRO.TIC	ARC009DAT	4	12	XX
DF	HYDRO.BND	ARC010DAT	1	16	XX
DF	BGEO5.PAT	ARC011DAT	11	16	XX

Figure 6.2 Partial INFO listing of workspace files.

Within the individual coverage directories, ARC/INFO stores both INFO files and the spatial files which are managed by the internal file management system. Referring to figure 6.1, the most common INFO files are

'tic' files, which contain identifiers and coordinates for coverage control points,

'bnd' files, which contain the coordinates of the minimum bounding rectangle of the

coverage,

'pat' files, which contain area, perimeter, and identifiers of polygons, and

'arc' files, which contain arc/node topology.

These INFO files contain binary, fixed length records. The data structure of these special files can be easily determined, either through the ARC/INFO documentation, or a query in INFO.

Some of the files in the coverage directory which store spatial data are

'arc' files, containing the coordinates of points in arcs, and

'pal' files, which record the arc/polygon topology.

There are also other types of files, such as coverage 'log' files which are stored in the coverage directories. These are discussed more fully in section 6.3.

6.3 Meta-data in ARC/INFO

Although lineage is not a feature of ARC/INFO GIS, the software utilizes a significant amount of meta-data. Most of this is for internal use, and not readily available to the user for other purposes. For example, digitizer registrations and rubber sheeting transformations provide - temporarily, on the computer monitor - estimates of RMS errors of the data after the operation. The ARC/INFO DESCRIBE and &DESCRIBE commands provide on-screen or in system variables, meta-data on the state of a coverage, and includes such data as whether or not polygon topology is present, the number of arcs in a coverage, and the geometric processing tolerances last used with a coverage.

ARC/INFO also provides data in the form of LOG files, WATCH files and AUDITTRAILs. These record in ASCII files the operations which have been performed, the coverages used, parameters used, etc., and so can provide a considerable amount of data about the input and output coverages and transformations of these which have occurred.

LOG files are automatically produced by ARC/INFO at the workspace and at the coverage level. These record command line entries of the user in INFO compatible file formats, and can be easily viewed by the user. Figures 6.3a and 6.3b give examples of workspace and coverage logs.

WATCH files are text files which contain a record of all of the non-graphic input and output of an ARC/INFO session. These files are in a special ARC/INFO format (see Figure 6.4a) and are most commonly used to produce AML files. The ARC/INFO command &CWTA will convert a watch file to an executable AML format. Figure 6.4b shows a converted watch file.

The AUDITTRAIL command is used in the ARCEDIT environment, and provides feedback to the operator on the changes which have occurred to the database in the current session. Figure 6.5 shows screen output of an ARCEDIT session with an audittrail being produced.

These types of files can provide users with a fair amount of data on the events which have taken place on the database; in a sense they provide a lineage of coverages, except of course for source manuscript documentation which are simply not part of the ARC/INFO data model. The disadvantage of these files is that they

```
a) Workspace log file

199007021444  19     6     0ae
199007021450   4    48     0ap
199007021451   0     2     0hpgl soils
199007021451   0     2     0hpgl soils soils.hp
199007021452   1     1     0hpgl bedrock bedrock.hp
19900703 920   0     1     0copy soils stest
19900703 920   0     1     0copy admin atest
19900703 921   0    16     0union stest atest otest 2.3
19900703 923   0    11     0clean otest otest2 3. 4. poly
199007161226   1    12     0intersect soils lakes soilslakes poly #

b) Coverage log file

199006081650   1    11     0generate soils82j
199007051427   1     4     0build soils82j poly
199007051513   0     0     0labelerrors soils82j
199007051540   1     4     0build soils82j poly
199007291435   0     0     0externalall
199008021346   0     0     0labelerrors soils82j
199008021354   1     8     0clean soils82j soils2 .001 .001 poly
199008021354   0     0     0labelerrors soils2
199008021413   1     8     0clean soils2 soils3 .001 .001 poly
199008021413   0     0     0labelerrors soils3
199008021413   0     0     0nodeerrors soils3
199008021418   1     9     0clean soils3 soils4 .001 .001 poly
199008021418   0     0     0labelerrors soils4
199008021418   0     1     0rename soils4 soils
```

Figure 6.3 ARC/INFO workspace and coverage log files, showing date and time, connect time, cpu usage, disk input/output (not applicable for Sun computers), and the ARC/INFO commands and parameters entered.

require the user to manually scan and interpret the files and determine the lineage, transformations, etc..

Audit and log files of transactions are also common in management information systems (MIS) environments because they help monitor transactions performed on a database, and therefore provide referential integrity (Grady, 1988). Two approaches to automating this procedure have been identified.

The first is a "front-end" or "pre-processor" approach, which is the more traditional (Grady, 1988). Any input to the system is trapped and checked against the

```
a) An ARC/INFO watch file

Arc: |> &r setup <|
/USR2/ROSSM/MSC
/usr2/rossm/msc/aml
9999,1,0,' '
Arc: |> display 9999 3 <|
Arc: |> arcplot <|
(C) 1988, 1989 Environmental Systems Research Institute, Inc.
    All Rights Reserved Worldwide
ARCPLOT Version 5.0.1

Arcplot: |> mapextent soilslakes <|
Arcplot: |> arcs soilslakes <|
Arcplot: |> mapextente * <|
Unrecognized command.
Arcplot: |> mapextent * <|
Arcplot: |> clear <|
Arcplot: |> arcs soilslakes <|
Arcplot: |> reselect soilslakes poly soilslakes-id = 16 <|
SOILSLAKES polys  : 1 of 45 selected.
Arcplot: |> polygonshades soilslakes 2 <|
Arcplot: |> quit <|
Leaving ARCPLOT...
Arc: |> &watch off <|

b) A watch file converted to an AML

&r setup
display 9999 3
arcplot
mapextent soilslakes
arcs soilslakes
mapextente *
mapextent *
clear
arcs soilslakes
reselect soilslakes poly soilslakes-id = 16
polygonshades soilslakes 2
quit
&watch off
```

Figure 6.4 a) A watch file and **b)** its AML equivalent.

rules of the data dictionary; if it is acceptable then the transaction is permitted and the entry is logged as lineage. Although it tends to slow down data entry it provides the best way to assure that data lineage is carried into the digital domain. In the context of a commercial GIS, where there is no data dictionary as such, this approach would require the trapping and interpretation of all input, before it reaches the application. This amounts to duplicating the command language of the software. In

```
Arcedit: audittrail full
There are 7 transaction(s)
Transaction 7: Arc 21 with User-ID 9 deleted
Transaction 7: Arc 105 with User-ID 9 added
----------
Transaction 7: Total arcs added 1, deleted 1
==========
Transaction 6: Arc 104 with User-ID 30 added
----------
Transaction 6: Total arcs added 1, deleted 0
==========
Transaction 5: Arc 26 with User-ID 7 deleted
----------
Transaction 5: Total arcs added 0, deleted 1
==========
Transaction 4: Arc 22 with User-ID 9 deleted
----------
Transaction 4: Total arcs added 0, deleted 1
==========
Transaction 3: Arc 28 with User-ID 8 deleted
----------
Transaction 3: Total arcs added 0, deleted 1
==========
Transaction 2: Arc 24 with User-ID 9 deleted
----------
Transaction 2: Total arcs added 0, deleted 1
==========
Arcedit:
```

Figure 6.5 An ARCEDIT AUDITTRAIL, showing detailed transactions on individual features.

the case of ARC/INFO, where there are hundreds of commands, this is a significant task. A pre-processor meta-database system to develop a lineage trail for error propagation used in conjunction with ARC/INFO GIS is briefly described by Lanter and Veregin (1990). In this approach, the pre-processor parses the command input and builds the GIS relationships. It then passes the command to the software. The disadvantage of this method is that software generated errors would not be trapped by the meta-interpreter, since they occur after they have passed through the parser. In addition, since the method of command entry to the software is limited to command line input, the use of ARC/INFO AMLs and menu systems cannot be accommodated.

The second approach is to apply a "back-end" check, or "post-process" the data. This has the advantage that it checks the contents of the database as opposed to the input, and therefore the interpretation of commands remains with the application. In addition, the full functionality of the software is available to the user. The disadvantage of this is that the integrity of the data base can be jeopardized. In this research, the concern is more for developing methods to manage lineage and study uncertainty than for database integrity issues. A post-processor approach to modelling lineage has therefore been adopted for UNCLE.

6.4 Modelling Lineage in UNCLE

6.4.1 Source Documentation

In the initial digitizing of the sample data set, the DIGIT files recorded the original digitizer registration errors and the coordinates of digitized points. In addition there are operator notes concerning the source, scale, etc. of data. In UNCLE these provide the source manuscript data for the various coverages and the beginning of the lineage trail.

As can be seen from the coverage log example for 'soils82j' in Figure 6.3, the very first entry indicates that the coverage was 'generated' from arcs in an 'ungenerate' file format. There is no provision in ARC/INFO for automatically recording source meta-data associated with the coverage. This is also true of directly digitized, or otherwise imported coverage features, coming into the ARC/INFO database. It is possible to enter comments into a log file, and view these at any time.

However, since the log files are in a fixed format for ARC/INFO use, and this file does not really further the development of automating the lineage procedure, this is of limited use.

The approach taken in UNCLE to include source documentation in the GIS database involves two steps. The first is to add a text file, at the coverage level, with the name of 'hist'. This file contains the original source documentation in its original form, plus any data which the operator wishes to add. This text file is easily edited and viewed from the operating system level. From within ARC/INFO it easily viewed by running a short AML called HISTORY which prompts the user for a coverage name and displays the text in a window; the operator can then scroll freely through the file. In UNCLE the DIGIT files for each coverage were copied to 'hist' files and the original coordinate values removed. Data on source, map scale etc., were then entered. An example is shown in Figure 6.6.

The next step taken to include source documentation as part of UNCLE was to create PROLOG facts and enter these into the PROLOG database. For each coverage, two facts are entered. One simply states that the coverage is an original coverage, and gives its name. The second fact incorporates in a more formal fashion the data from the 'hist' file for that coverage. Figure 6.7 gives examples of facts for source documentation.

6.4.2 Establishing the Lineage Database

The lineage database in UNCLE is derived from the coverage LOG files and

```
!
!soils82j.out
!
!    FILE OPENED ON 24-MAR-89 AT 10:48:56.
!
!***** MAP MOUNTING INFORMATION *****
!
!    MAP NAME: soil82j            MAP UNITS: METRES
!
! Point ID    Northing       Easting       X-Dig     Y-dig     Res-X    Res-Y
!  LL       5538890.907    571668.795      177.7     134.8     0.42     0.13
!  UL       5650082.180    570171.205      184.9     576.0    -0.43    -0.13
!  UR       5653890.699    710495.889      741.9     575.4     0.43     0.13
!  LR       5542725.237    714991.281      748.3     134.5    -0.42    -0.13
!
!                                                    RMS       0.42     0.13
!                                              TOTAL RMS            0.31
!
!    SCALE FACTORS      NORTH  252167.972     EAST  251636.109
!    OFFSETS            NORTH 5503636.196     EAST  527812.907
!    ROTATION                 1.60 Deg.
!    NON-PERPENDICULARITY    -0.01 Deg.
!
!
! source map - Canada Land Inventory, Soils Capability for Agriculture
! scale 1:250000
! converted with UC-DSE DIGIT program
! operator - Ross Miller
```

Figure 6.6 An UNCLE 'hist' file, showing original information from DIGIT program, and incorporating operator comments. These can be displayed from within ARC/INFO in a 'pop-up' screen.

therefore builds lineage based on ARC/INFO commands and arguments which have been executed. Lanter (1990) describes a LISP based lineage program, which also uses knowledge of the commands and arguments, but does this before the execution of the commands.

In UNCLE, A UNIX script (LOG2PRO) and an AWK program (L2P.AWK) were written to reformat the LOG file entries into PROLOG facts. These programs reside at the workspace level; the PROLOG files reside at the coverage level. Figure 6.8 shows the PROLOG file corresponding to the coverage log shown in Figure 6.3b. PROLOG files for each coverage were produced and then entered into the

```
/* source coverage facts */

original(sgeo5).
original(bgeo5).
original(soils82j).
original(kvadmin).
original(hydro).
original(cadastral).
original(kvlakes).

/* source coverage info */

/* source_doc(cover, source, scale,  method, operator).*/

source_doc(sgeo5,nts,250000,manual_digitizing, ross_miller).
source_doc(bgeo5,nts,1000000,manual_digitizing, ross_miller).
source_doc(soils82j,cli,250000, manual_digitizing, ross_miller).
source_doc(kvadmin,alberta_enr,100000, manual_digitizing, ross_miller).
source_doc(hydro,nts,50000, manual_digitizing, canny_mulaku).
source_doc(cadastral,alberta_doh,32000, manual_digitizing, canny_mulaku).
source_doc(kvlakes, nts,50000, manual_digitizing, canny_mulaku).
```

Figure 6.7 PROLOG facts for documentation on source manuscripts. Text between '/*' and '*/' are comments.

```
generate(soils82j).
build(soils82j,poly).
labelerrors(soils82j).
build(soils82j,poly).
externalall.
labelerrors(soils82j).
clean(soils82j,soils2,0.001,0.001,poly).
labelerrors(soils2).
clean(soils2,soils3,0.001,0.001,poly).
labelerrors(soils3).
nodeerrors(soils3).
clean(soils3,soils4,0.001,0.001,poly).
labelerrors(soils4).
rename(soils4,soils).
```

Figure 6.8 PROLOG Facts for the 'soils' coverage, after conversion from ARC/INFO LOG files. Derived from the LOG file shown in Figure 6.3b. Commands become functors, and coverage and parameters become arguments.

PROLOG database. Figure 6.9 shows part of the rule base for the sample data set. This method of converting LOG files to PROLOG facts works adequately for transactions which occur in the ARC module of ARC/INFO. However, there are some transactions which are recorded in the LOG file but which involve additional transactions in other modules. For example, the ARCEDIT module can be executed from ARC, and an entry will be placed in the LOG file indicating the module called. However, the editing operations which are subsequently performed in ARCEDIT are not recorded in the coverage LOG. Consequently, the prototype currently creates PROLOG facts only for those LOG entries whose transactions take place entirely in the ARC module.

In order to determine the source documentation of a coverage in UNCLE, it is necessary to write PROLOG rules about the relationships of coverages. These rules take the form 'source(cover)', where 'cover' is the name of the coverage of interest to the user.

Source coverages are declared as such using the 'original' predicate in the database, and source documentation for these is declared in the 'source_doc' predicate.

Finding the 'source' of an original coverage and its source documentation is trivially programmed, since 'original' and 'source_doc' are simple facts in the database. This is programmed as

```
source(X,X):-original(X).
```

In non-PROLOG terms, this declares "The source for any coverage X is itself, if X

is an original coverage". PROLOG would then simply have to match the query

? - source(soils82j,X).

with the database facts. In this case since 'soils82j' is 'original' and has 'source_doc' (Figure 6.9) then the query would succeed, and the source documentation would then be printed automatically. An example of such a query from an interactive session with UNCLE is given in Figure 6.10.

```
/* admin boundary facts */

create(kvadmin).
generate(kvadmin).
clean(kvadmin,admin_clean,0.001,0.001,poly).
labelerrors(admin_clean).
build(admin_clean,poly).
labelerrors(admin_clean).
rename(admin_clean,admin3).
externalall.
clean(admin3,admin2,0.001,0.001,poly).
rename(admin2,admin).


/* hydro facts */

generate(hydro).
externalall.

/* lakes facts */

create(kvlakes).
generate(kvlakes).
build(kvlakes,poly).
labelerrors(kvlakes).
rename(kvlakes,lakes).
externalall.

/* test data for polygon overlay */

union(soils,lakes,soilslakes).
```

Figure 6.9 A portion of the PROLOG rule base in UNCLE

However, to automatically determine which coverages are derived from other coverages, i.e. lineage, a different approach is needed. An automated solution can be modelled after a manual approach. For example, to manually determine the lineage of 'soils2' in ARC/INFO, one would examine the LOG file of Figure 6.3. This log

```
?- source(soils82j,X).

soils82j is an original coverage

        soils82j
                source : nts
                scale : 1 : 250000
                method : manual_digitizing
                operator : ross_miller

X = soils82j

Yes ;

No

?-
```

Figure 6.10 An interactive session with UNCLE, showing how source documentation can be retrieved for an original coverage.

contains the entry

'199008021354 1 8 0clean soils82j soils2 .001 .001 poly'

which demonstrates that lineage is implicit in the 'clean' command. There is no explicit declaration that 'soils2' is derived from 'soils82j'. However, if one knows what the ARC/INFO commands and the arguments mean, then it is possible to manually trace back from a derived coverage to an original coverage. In this manner, one could establish the lineage of events. Additional information such as source documentation could then also be manually looked up. This manual process of interpreting the commands or facts, and their parameters, requires knowledge of the transformations and therefore relationships of the input and output coverages. For a small number of transformations, such a manual approach is workable; however, an automated approach is necessary for more complex situations. In order to automate this, it is necessary to formalize this knowledge.

UNCLE initially has no knowledge of such relationships or transformations.

It is necessary to write rules about these in a fashion similar to the PROLOG program example given in Chapter 4. To determine lineage, it is necessary to create the equivalent of the 'connect' clauses described in Chapter 4, which were used there to define a particular kind of spatial relationship between points. Similarly, the lineage clauses are used in UNCLE to define a number of different types of transformation relationships between coverages. These clauses utilize the ARC/INFO commands which are recorded in the LOG file and brought over to the PROLOG database as predicates. The coverage names and parameters in parentheses of the PROLOG database are the arguments for these transformations. These form facts for all of the ARC/INFO coverages.

One can view the lineage of the coverages as a family tree. Each derived coverage has one or more 'parent' coverages, and each of those has a 'parent', and so on, back to some 'original' or 'source' coverage. It is possible to model this tree in PROLOG by encoding the relationships as clauses and applying recursive programming techniques to these.

For example, the original coverage 'soils82j' was cleaned to coverage 'soils2'; the PROLOG fact is

clean(soils82j, soils2, 0.001, 0.001, poly).

as shown in Figure 6.9. Here the relationship is 'clean' because 'soils82j' is transformed to produce 'soils2'; coverage 'soils82j' is therefore (implicitly) an ancestor of coverage 'soils2'. In PROLOG it is possible to declare this relationship with the general rule

```
ancestor(X,Z):- clean(X,Z,_,_,_).
```

meaning that 'A coverage X is an ancestor of a coverage Z if X has been cleaned to produce Z'. This can then be incorporated into the 'source' predicate as

```
source(X,Y):-
        ancestor(Y,X),
        original(Y).
```

meaning 'The source for coverage X is coverage Y if Y is an ancestor of X and Y is an original coverage'. If the query

```
?- source(soils2,Y).
```

were submitted, PROLOG would call the 'ancestor' clause above, then call the 'clean' fact in the tail of that rule, find that 'clean(soils82j,soils2, 0.001, 0.001, poly)' in the database was true, and that 'original(soils82j)' was true. The query would therefore succeed, and the source documentation for 'soils82j' would be printed. This example is shown in Figure 6.11.

This first rule for ancestry will work adequately for relationships at the top of the tree, for a parent and a child. However, for additional generations it is necessary to employ recursion. The following rule adds recursion to the first ancestor clause;

```
ancestor(X,Z):-
        clean(X,Y,_,_,_),
        ancestor(Y,Z).
```

and makes it possible to determine lineage to any depth.

In a similar fashion, the transformation relationships between coverages can be declared for each command in the LOG file. So for example, a nominal transformation such as renaming a coverage would be declared as;

```
?- source(soils2,X).

soils2 is derived from original coverage soils82j

        soils82j
                source : nts
                scale : 1 : 250000
                method : manual_digitizing
                operator : ross_miller

  X = soils82j

Yes ;

No

?-
```

Figure 6.11 Example of a PROLOG query in UNCLE to determine the lineage across one or more 'generations' of coverages.

```
ancestor(X,Y):-rename(X,Y).
ancestor(X,Z):-rename(X,Y),ancestor(Y,Z).
```

In UNCLE this has been done for all the commands executed in ARC/INFO for the sample data set.

The determination of lineage described above involves a straightforward line of lineage, from a single 'parent' to several 'child' coverages. However, in GIS software, many of the operations involve two or more coverages, so a descendent coverage can have multiple ancestors. For example, in ARC/INFO the UNION and INTERSECT commands are used to overlay two input sets of coverage features in logical OR and AND operations respectively, to produce a third, output set. This output set can then be used in further operations, thereby increasing the complexity of the lineage trail. However, lineage can still be determined in these cases by adding the necessary and correctly formulated rules for these relationships. UNCLE will

successfully find all of the 'ancestor' solutions for a coverage, regardless of the manner in which the coverage evolved. This powerful capability of UNCLE is critical in order to study uncertainty through a series of GIS transformations.

UNCLE works only on the facts and rules existing in the database; these facts and rules define the knowledge domain of UNCLE. If other operations were performed in ARC/INFO with different arguments, it would be necessary to bring those facts into the database, and also add the corresponding rule which makes the new relationship known to UNCLE. Since PROLOG databases can be easily modified (as was shown in Chapter 4), it is possible to expand the domain of knowledge relatively quickly.

For example, the derived coverages 'soils' and 'lakes' were combined together with the UNION command. This command performs a logical OR with the sets of input features and produces an output coverage set which contains all of the features in the originals. This output coverage in UNCLE is called 'soilslakes'. In order to maintain the lineage capabilities of the rule base, a relation is defined with 'union' and 'ancestor' in a similar fashion to the other commands mentioned above. If the goal

```
?- source(soilslakes, X).
```

is then submitted, UNCLE will respond with the results shown in Figure 6.12.

As was demonstrated in Chapter 4, it is possible to submit queries to PROLOG in a number of ways, and PROLOG will perform searches of the entire database to satisfy these. This introduces a great deal of flexibility in UNCLE. For

```
?- source(soilslakes,X).

soilslakes is derived from original coverage kvlakes
        kvlakes
                source : nts
                scale : 1 : 50000
                method : manual_digitizing
                operator : canny_mulaku

X = kvlakes

Yes;

soilslakes is derived from original coverage soils82j
        soils82j
                source : cli
                scale : 1 : 250000
                method : manual_digitizing
                operator : ross_miller

X = soils82j

Yes;

No

?-
```

Figure 6.12 Determining lineage over several generations, with several source coverages.

example, it may be of interest to find all of the derived coverages of an original coverage, rather than the original sources of a derived coverage. The query submitted might be;

?- source(Z,soils82j).

meaning 'What coverages exist for which 'soils82j' is the source coverage?'. PROLOG would respond with the solutions shown in Figure 6.13. These solutions give some idea of the nature of the lineage. However it is still not completely clear exactly how all these coverages are related.

This last point raises an important issue. In many cases it is important to know how - in what order, and by what relationships - the lineage of a coverage has

```
?- source( Z, soils82j ).
  Z = soils82j
Yes ;
  Z = soils
Yes ;
  Z = soilslakes
Yes ;
  Z = soils2
Yes ;
  Z = soils3
Yes ;
  Z = soils4
Yes ;
No
?-
```

Figure 6.13 Finding all of the descendants of a coverage.

evolved. Although these data are determined and used internally when PROLOG is solving a goal, it is not available to the user - the only visible evidence is the final output. But this kind of information can be important for the user in the day to day operation of a GIS, especially in order to determine the "fitness for use" of the output data. It is also required for studying additional aspects of uncertainty. This issue is addressed in the next section.

6.4.3 Meta-Level Programs in UNCLE

Meta-programs are programs which treat other program as data, and can analyze, transform or simulate these. One particular class of meta-programs are

meta-interpreters, which is an interpreter for the language which is written in the language itself. Meta-interpreters give access to the computational process of the language (Sterling and Shapiro, 1986). Due to its symbolic manipulation capabilities, PROLOG is a particularly powerful language for meta-programming and therefore for developing meta-interpreters (Bratko, 1990).

One example of a meta-interpreter is a 'tracer' program, which simulates the standard PROLOG debugging facility and therefore simulates the computational model of PROLOG programs. Debugging a PROLOG program can be especially confusing, because of the highly recursive nature of the language, and the large amount of backtracking which occurs. A standard feature of PROLOG is a trace facility which shows the many calls and pattern matching of clauses which occur as the program executes. Tracers provide detailed information for programmers, but produce far too much information for ordinary users. Moreover, they are still not usable by the programmer to include as code. However it is comparatively easy for a programmer to write a tracing meta-interpreter in PROLOG which achieves the same results. (Bratko, 1990; Sterling and Shapiro, 1986).

An example of a more sophisticated and useful meta-interpreter, which applies concepts similar to the 'tracer', is one which generates a "proof-tree" for a goal which is submitted to PROLOG. A proof-tree is an explanation of how the final conclusion follows from the rules and facts in the knowledge base.

In UNCLE a meta-interpreter to generate a proof tree for lineage generation has been implemented. This meta-interpreter uses the predicate 'show_lineage_for'

and takes two arguments. The first argument is any descendant coverage, and the second is any ancestor coverage in the tree, up to the original coverage. An example for the query

?- showlineage(soils, soils82j).

is shown in Figure 6.14. The proof-tree in the figure shows exactly how the lineage developed from the ancestor to the descendant coverage, including all the arguments

```
?- showlineage(soils,soils82j).

derived_from(soils,soils82j) is true because

 IF ancestor(soils82j,soils)
THEN lineage(soils,soils82j)

ancestor(soils82j,soils) is true because

 IF clean(soils82j,soils2,1.00e-03,1.00e-03,poly) AND ancestor(soils2,soils)
THEN ancestor(soils82j,soils)

clean(soils82j,soils2,1.00e-03,1.00e-03,poly) is a fact.

ancestor(soils2,soils) is true because

 IF clean(soils2,soils3,1.00e-03,1.00e-03,poly) AND ancestor(soils3,soils)
THEN ancestor(soils2,soils)

clean(soils2,soils3,1.00e-03,1.00e-03,poly) is a fact.

ancestor(soils3,soils) is true because

 IF clean(soils3,soils4,1.00e-03,1.00e-03,poly) AND ancestor(soils4,soils)
THEN ancestor(soils3,soils)

clean(soils3,soils4,1.00e-03,1.00e-03,poly) is a fact.

ancestor(soils4,soils) is true because

 IF rename(soils4,soils)
THEN ancestor(soils4,soils)

rename(soils4,soils) is a fact.
Yes ;
No

?-
```

Figure 6.14 A proof tree in UNCLE which shows the relationships between coverages and how they are related in the lineage.

in the facts and rules.

Meta-programs are also used to help model uncertainty in expert and rule-based systems. The previous examples have demonstrated a type of true-false knowledge. If the facts and rules were in the domain of the database, and the proper query was submitted, PROLOG would 'prove' it to be true. If something was not 'proven' by UNCLE does not mean it was false, only that it does not exist in the database. However, it has long been recognized that the propositions or hypotheses are not true or false but may have varying degrees of certainty of confidence, which should be taken into account.

A PROLOG program with certainty or confidence factors is one which associates a numerical measure of certainty with the individual clauses (facts or rules) in the database. These are encoded in a form such as 'cwcf(clause,cf)', where 'clause' represents the facts and rules, and 'cf' is a numerical measure of some sort. These are then embedded in clauses with the functor 'cwcf', and the facts or rules and their associated factors can be manipulated together as data. The choice of 'cwcf' (short for 'clause with certainty factor') is completely arbitrary, having semantic importance only to the programmer; it has only symbolic importance in PROLOG in the pattern matching process.) In order to process these 'meta-clauses' a meta-level program is required. This is similar in many respects to the 'proof-tree' meta-interpreter. However, in addition to analyzing and using the PROLOG program itself, the meta-program adds rules for computing uncertainty measures. By doing so, it is possible to propagate these measures through a series of logical propositions. In the case of

UNCLE and the GIS data set, it provides a promising means of tracking the error or uncertainty through the transformations introduced by the GIS software.

For example, the UNION of 'soils' and 'lakes' produces a third coverage ('soilslakes') consisting of features from both of the input coverages. These two coverages were selected to test the propagation of uncertainty in the UNCLE database, however, since relationships can be determined in UNCLE for any sets of coverages, others might also have been used. To estimate the accuracy or uncertainty of a derived coverage, a conservative approach might be to take the least accurate input coverage value and assign it to the output coverage. This simple model is used for this example.

In UNCLE clauses were assigned a 'certainty factor' (cf) between 0 and 1. In rule-based systems these are often ad-hoc measures, although they may also be probabilistic in origin (Bratko, 1990). The values used in this example for the source maps are subjective and represent a crude measure of the relative accuracy of the original maps. These are based on the contents of the 'hist' file. For the 'original' coverages, values were assigned to the clauses. For other clauses, this is done in a general fashion, with all clauses sharing the same value. For example, the 'rename' predicate, which performs a nominal change to a coverage, is assigned a value of 1 in a generic clause. All predicates and their corresponding factors were changed to a form (i.e. data) where they can be treated as part of a 'cwcf' clause.

In order to determine the minimum 'certainty factor' of the output coverage, a general purpose meta-program was written. In this example, the program works by

first finding the 'lakes' lineage of the 'soilslakes' coverages and acquires the initial value for 'kvlakes'. It then backtracks through the lineage applying the intermediate certainty factors associated with the transformations, simultaneously updating the value with the minimum value encountered. This process is repeated for the 'soils82j' coverage. The final value of the certainty factor for 'soilslakes' is determined by using a PROLOG rule to find the minimum value of the two values derived from the two lineages. Figure 6.15 shows how these are implemented, and how these are presented to the user by submitting a goal to UNCLE.

```
?- findcf(soilslakes).

?-soilslakes is derived from original coverage kvlakes
        kvlakes
                source : nts
                scale : 1 : 50000
                method : manual_digitizing
                operator : canny_mulaku
soilslakes is derived from original coverage soils82j
        soils82j
                source : nts
                scale : 1 : 250000
                method : manual_digitizing
                operator : ross_miller
minimum cf for coverage is 5.000000000000000e-01Yes ;
No

?-
```

Figure 6.15 Computing uncertainty in UNCLE.

In order to support the interval valued approaches suggested previously (Section 3.3.2), the single valued certainty factors can be replaced by the same notation shown in Table 3.1. This is the approach taken by such expert systems as PROSPECTOR (Bratko, 1990). PROSPECTOR employs 'subjective probability

measures' in the form '[S,N]' where S denotes the support and N the necessity for a given hypothesis.

In UNCLE, this method of representation is accomplished by employing a PROLOG list for the pair of certainty factors. The 'cwcf' clauses were changed to the form

cwcf(clause,[S|N])

where 'clause' represents the UNCLE facts and rules, and '[S,N]' denotes some interval of uncertainty. To determine a resulting interval through a series of transformations, the rule base tracks the minimum of the support and the maximum of the necessity values. An example is shown in Figure 6.16.

The previous examples have used predetermined, arbitrary numeric measures of uncertainty associated with the clauses. However uncertainty may be measured on

```
?- findcf2(soilslakes).

soilslakes is derived from original coverage kvlakes
        kvlakes
                source : nts
                scale : 1 : 50000
                method : manual_digitizing
                operator : canny_mulaku

soilslakes is derived from original coverage soils82j
        soils82j
                source : cli
                scale : 1 : 250000
                method : manual_digitizing
                operator : ross_miller
minimum cf for coverage is 5.000000000000000e-01maximum cf for coverage is 9.000000000000000e-01Yes
;
No

?-
```

Figure 6.16 Computing intervals of uncertainty.

other types of scales, such as nominal or ordinal. Hinde (1986) shows how fuzzy representations and grades of memberships can both be incorporated into PROLOG. UNCLE could also be adapted to include this type of representation. This could be used for example, with linguistic descriptions of map quality or attribute data.

It is necessary to provide methods of computing uncertainty measures at execution time, rather than just assigning predetermined values. This is important because the effect of a transformation on the overall uncertainty measure may be a function of the data used in the transformation itself. For example, in the absence of user-specified geometric tolerances for vertex coalescence in the UNION command, ARC/INFO sets the value based on the geographic extent of the coverages, as 1/10000 of the extent of the coverage. At execution time it is required to determine this extent, and apply it.

Different methods of computing uncertainty can be incorporated into UNCLE by adding the appropriate rules, and entering the corresponding uncertainty measures or the computations to generate them. For example, in order to arrive at some estimate of the relative positional accuracy of a composite of map features, a simple test was performed in UNCLE. Positional error can result from errors in the source document, from digitizing, and from subsequent geoprocessing operations. Certainty values representing positional accuracy of lines in the original coverages were determined based on a number of factors (see Table 6.1). The CMAS values were based on original map scales (for 1:50000 and 1:250000 scale maps) and taken from table of circular map accuracy standards (CCSM, 1984). Values for 1:1000000 and

1:20000 were linearly extrapolated from the tables. The tolerance values were based on those used in ODYSSEY processing; digitizer registration values were extracted from DIGIT files, and average values of operator digitizing accuracies were based on findings from Bolstad et al (1990). For the transformations undertaken in ARC/INFO, similar measures were determined. Some ARC/INFO transformations are nominal, for example, a name change by a RENAME command. For these types of transformations, values of zero were assigned to the 'cwcf' rule for that transformation. For ARC/INFO transformations which add positional uncertainty, values were either determined or computed, and the 'cwcf' rules were modified accordingly. For example, the CLEAN transformations used tolerances of 0.001 metres. This means that any point in the coverage may be moved by that amount. This value was included in the rule base as part of the CLEAN relations, after conversion with LOG2PRO. The UNION command utilized tolerance values based on the geographic extent ('bnd') of the coverage. The default value tolerance for coverages is 1/10000 of the width of the 'bnd'. Rules were added to the data base which computed these values. The 'cwcf' clauses for CLEAN were modified to call these new rules and compute at execution time the amount of uncertainty contributed from this transformation.

These errors were treated as independent, and an estimate of positional accuracy of the final digital product is determined as the sum of the individual errors. This type of positional error is an example of the 'epsilon band' concept of error in cartographic lines proposed by Chrisman (1982). The epsilon band represents a zone

Coverage	CMAS	Registration	Operator	Tolerance (ODYSSEY)	Tolerance (CLEAN)	BND/ 10000
soils82j	250	77.5	13.4	75.0	0.001	3.1
sgeo5	250	32.5	13.4	250.0	0.001	2.9
bgeo5	1000	100.0	53.5	400.0	0.001	3.3
kvadmin	100	9.8	1.1	100.0	0.001	8.4
kvlakes	50	6.5	2.7	0.109	n/a	0.5

Table 6.1 Values of positional uncertainty used in overlay operation. All values in metres.

of uncertainty surrounding a line; the 'true' position of the line may fall anywhere within the zone.

Rules were added which modelled the final positional uncertainty as the sum of the original and all intermediate values. This model was applied to the coverage 'soilslakes', which was derived from original coverages 'soils82j' and 'kvlakes'. Using the lineage tracking features of UNCLE, uncertainty measures were acquired for the original coverages 'soils', and that value was then propagated through the various transformations. A value from 'lakes' was determined in the same manner. The result was a pair of values, indicating a rough measure of the positional accuracies of the linework of the two input coverages. The sample output for this test is shown in Figure 6.17, and indicates that the positional uncertainty associated with the 'lakes' coverage (derived from 1:50000 scale maps) is much less than that from 'soils' (1:250000).

For relatively simple computations, new rules can be added and associated

```
?- findcf3(soilslakes).

soilslakes is derived from original coverage kvlakes
        kvlakes
                source : nts
                scale : 1 : 50000
                method : manual_digitizing
                operator : canny_mulaku
soilslakes is derived from original coverage soils82j
        soils82j
                source : cli
                scale : 1 : 250000
                method : manual_digitizing
                operator : ross_miller
estimated cf for coverage is [5.98090e+01,4.16403e+02]
Yes
```

Figure 6.17 Output from UNCLE showing positional uncertainty propagated during processing. 'CF' values in metres.

with the transformations as was done above with the 'bnd' values. However since PROLOG is poorly suited to handle numerical manipulation, it is also necessary to have the capability to perform more complex analyses associated with types two and three uncertainty, eg. statistical methods. This can be done in two ways. Both involve the use of built-in features of BIMprolog.

The first method is to make use of predicates which allow temporary exit to the UNIX operating system, and execution of UNIX commands, scripts, or user programs. In this method, the rules which associate transformations with certainty factors could be extended to include these built-in predicates. When these new rules are encountered, execution would pass to the operating system. When the operating system task(s) have been completed, execution returns to the rule base. In this method, no data is passed directly between the rule base and the operating system, and intermediate files would have to be employed. This entails modification of the

rule base, to add file input and output rules. This method has been tested to update the PROLOG facts in the file system, by first exiting to UNIX, executing LOG2PRO on selected coverage LOG files, and then returning to UNCLE. Further work is required to automate this procedure further and provide full updating of the UNCLE rule base.

The second method uses external language interfaces. BIMprolog provides external language interfaces, with parameter passing, to C programs. The rules which associate transformations with certainty factors could be extended to include rules which invoke the user's external programs. When these rules are encountered in goal seeking, execution would pass, along with the parameters, to the external program. On completion, the results could be passed back into the program.

6.5 Utilizing Spatial and Non-spatial Data in UNCLE

Thus far, UNCLE has dealt with data at the coverage level. Source documentation, lineage, and simple calculations of certainty measures are developed for original, derived, and composite coverages. Although these capabilities may provide a sufficient basis for many users to deal with uncertainty in their system, it does not satisfy all of the requirements listed in Chapter Four. Access to spatial and non-spatial data at the feature level, i.e. to points, lines, polygons and their associated attributes, is also required.

In ARC/INFO features are stored in INFO and internal file systems, and linked by system generated identifiers. These files are binary, either fixed or variable

length, and their structure may be documented or undocumented. These can be accessed through direct reading and writing of files, or through some translation method.

6.5.1 Direct Reading and Writing of ARC/INFO Files

The most desirable situation for certain users of GIS software, such as researchers and software developers, is to be able to perform direct reading and writing of all GIS files with user programs. This provides full access to the spatial and non-spatial definitions of features (eg. arcs, nodes, polygons) in the database. However this is very undesirable for the software company, who have a considerable vested interest in keeping this information confidential from their competitors. Consequently this information is not readily available, and in ARC/INFO, the file structures, especially of the internal files, are difficult to determine. This is made more difficult because GIS data models and their implementation in file systems tend to be fairly complex, in order to manage and maintain the large amounts of data, the organization of coverages and layers, and all of the linkages and spatial relationships among these. In ARC/INFO for example, there are a number of binary cross-reference files maintained by the system which specify the topological relationships of nodes, arcs, and polygons.

In ARC/INFO, some insight into the file system organization can be gained by studying the software documentation, which provides conceptual views of the data model. However, these do not provide sufficient data to understand the binary file

structures.

Another potential source of information is other, similar software. It has been previously noted that ARC/INFO was partially based on the ODYSSEY GIS. Upon examination it was determined that the chain model of ODYSSEY and the arc/node/polygon topology model of ARC/INFO are very similar. Using the various CDB and LDB translator programs of Chapter Five as a basis, the C program RDARCS was developed. This program successfully reads an ARC/INFO binary ARC file. However, it is not complete, as there are several additional fields in the ARC file whose purpose and contents have not been determined. Consequently it is currently of limited use. Because of the many interrelated files in ARC/INFO, the difficult trial and error approach to deciphering these, and because the internal file system may change from time to time with new releases of software, it is doubtful that pursuing direct reading of binary internal files is worthwhile for UNCLE.

INFO files are also stored in binary format. However, the record and field lengths of these can be determined from within INFO, and user programs can be written to access these, as long as one knows the full UNIX names and locations of these in advance. However, INFO manages files through the combination of binary 'arcdr9', 'arc*nnn*dat' and 'arc*nnn*nit' files, whose contents are known but whose structures are unavailable. As a result these would also have to be deciphered by the user. Although this is much easier than deciphering the spatial files, it is likewise doubtful that for the purposes of UNCLE this is worthwhile.

6.5.2 File Translation Methods

A more suitable approach to accessing data at the feature level is to translate the data to an easily readable format, and then develop user programs to read and write these. This is possible for both internal spatial files and INFO files in ARC/INFO.

ARC/INFO internal files can be easily exported to a generic ASCII format (UNGENERATE), and to a number of industry or government formats, including AUTOCAD DXF, and USGS DLG. In some of these a significant amount of information is lost. For example, all topology is lost in UNGENERATE and DXF format. The most suitable format for use in UNCLE is the DLG format. This format maintains all arc, node and polygon topology, and also provides the coordinates for features. Since the programs DLG2LDB and CDB2DLG were previously written to translate to and from this format in order to develop the database (see Figure 5.2), little modification is required to produce files which can be read into UNCLE, or used in external programs.

INFO binary files can be converted to ASCII files by changing the default spool device and redirecting the output of an INFO LIST command to a file.

Although file translation methods add an additional step in the process of using feature level data in UNCLE, they are easy to use, require less development time and are independent of changes to internal file organization of the software. These methods are therefore more suitable for use in a prototype system and consequently have been adopted for UNCLE.

6.5.3 Experimentation with Feature Level Data

An experiment was performed which applied UNCLE rule-based reasoning about uncertainty to a common problem in GIS. This is the problem of inter-layer consistency of data sets, which results in 'spurious' ('sliver') polygons. Polygon data associated with the 'soilslakes' coverage was used for this.

Spurious polygons occur when two coverages are combined by a GIS transformation, such as the ARC/INFO UNION command. Spurious polygons are typically small, elongated polygons which are produced when two representations of the same geographic feature, from two different sources, are combined during geoprocessing. In the 'soilslakes' example, both input coverages contain polygon representations for the Kananaskis Lakes. Figure 6.18 shows the coverages 'soils' and 'lakes'. When the polygons in these coverages are combined, a number of spurious polygons were produced. Figure 6.19 shows graphically the result of such an operation. Similar problems occurred with other combinations of coverages in the data set. For the administrative and geological coverages, for example, a graphical overlay produced numerous spurious polygons along the inter-provincial boundary. This inconsistency occurs because of positional uncertainty in the data sets, which may in turn be caused by differences in source document scale or accuracy, data capture method, operator error, etc..

The problem of spurious polygons is common and has plagued GIS processing for several years. They have several negative effects. They can be numerous, accounting for a large percentage of all the polygons produced in an overlay

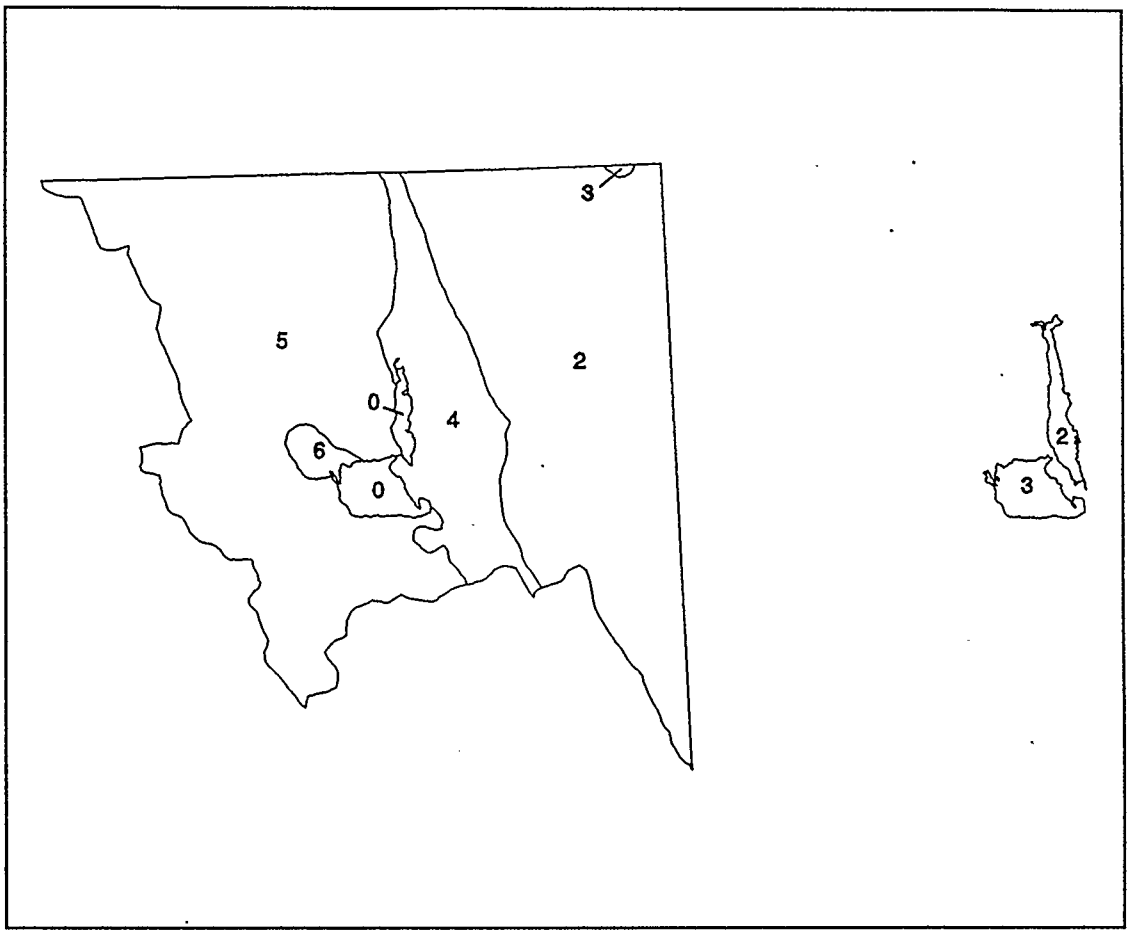

Figure 6.18 Polygons for 'soils' and 'lakes' coverages. Numbers shown indicate polygon identifiers. Enlargement of shaded study area shown in Figure 5.1.

operation. As can be seen from Figure 6.19, spurious polygons also contribute to visual clutter in the map product. If stored in the database they will consume disk space, and will affect processing efficiency in subsequent operations. Perhaps most importantly they affect the results of GIS analysis, and consequently the value of that information.

Spurious polygons can be removed by either manual or automatic methods. Manual methods require an operator to somehow define what constitutes a spurious

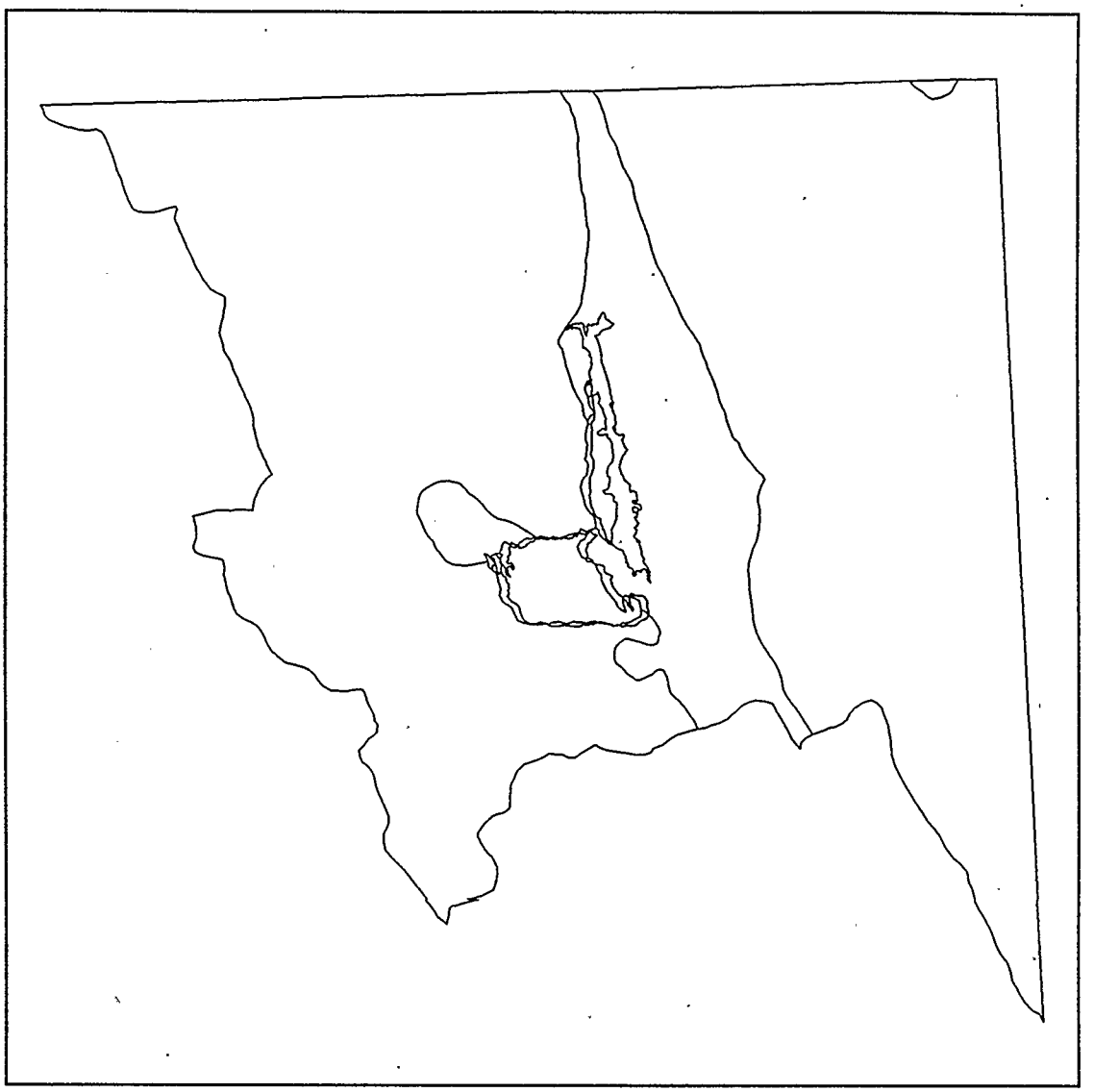

Figure 6.19 Overlay of 'soils' and 'lakes' coverages. Enlargement of shaded area shown in Figure 5.1.

polygon, and to identify these in the database and interactively edit their component linestrings. This may require a large amount of zooming, panning, selection and deletion of graphic elements. It can be a very tedious and time consuming task, particularly where the number of polygons is large. In establishing a database, in map

production, or to correct coverages for analytical purposes, this can be very costly. However, a manual approach has the advantage that the operator can make intelligent decisions. Factors such as familiarity with meta-data about the input coverages, instructions (rules) from his or her superior or the end purpose of the data may guide the operator in his or her work. In other words, the corrective action required is dependent on the application and context.

Automated methods typically offer two approaches to remove spurious polygons. The first technique allows for a user-defined geometric tolerance to be applied during the overlay operation. This specifies a distance between points, below which points and lines will be automatically eliminated. The proper use of this technique therefore requires that the operator have some a priori knowledge of the characteristics of the data set, including the distance between features. Misuse of tolerance values can have unpredictable or undesirable results. For example, by setting the value too large, distant features may also be merged and disappear entirely from the output coverage.

The second automated technique is to process the output data set. In ARC/INFO for example, the ELIMINATE command allows the user to first select polygons based on any INFO items, and then delete these from the database. The polygons are eliminated by deleting the longer of the component arcs of the polygon. For example, in the 'soilslakes' coverage shown in Figure 6.19, all polygons having an area less than 500000 square metres were selected and eliminated. The result is shown in Figure 6.20 with the original 'lakes' coverage superimposed. This is an

example of a trivial solution to the problem, which can contribute to an increase in uncertainty in database. This is because no consideration is given to the positional accuracy of the lines. An examination of the 'hist' files for the coverages reveals a significant difference in the scale and accuracy of source documents for 'soilslakes', with the 'kvlakes' polygons considerably more accurate than the 'soils82j' polygons. In a manual solution, the operator may be aware of this or have access to this knowledge and intelligently solve the problem by selectively eliminating the less accurate data, i.e. the soils data, and therefore reducing uncertainty in the database.

The spurious polygon problem is also a consideration in building a GIS database. It is not only important to have consistency in individual layers or coverages, but to ensure that the features which occur in more than one coverage are identical. Achieving (and maintaining) this inter-layer consistency is an aspect of the larger problem of data validation and data integrity. Most commonly this is done by manual methods, by graphical overlay and editing. It can therefore be an expensive and lengthy procedure.

It was decided to investigate how UNCLE could be used to help establish inter-layer consistency in a database. UNCLE can provide much of the data that an operator might need to resolve the problem. For example, the rule base contains meta-data about the coverages, and relationships between derived and ancestor coverages can be determined. Additional rules can be incorporated into the program which might reflect the parameters for identifying and removing the polygons. Access to feature level data may guide the operator in their selection of the line work to be

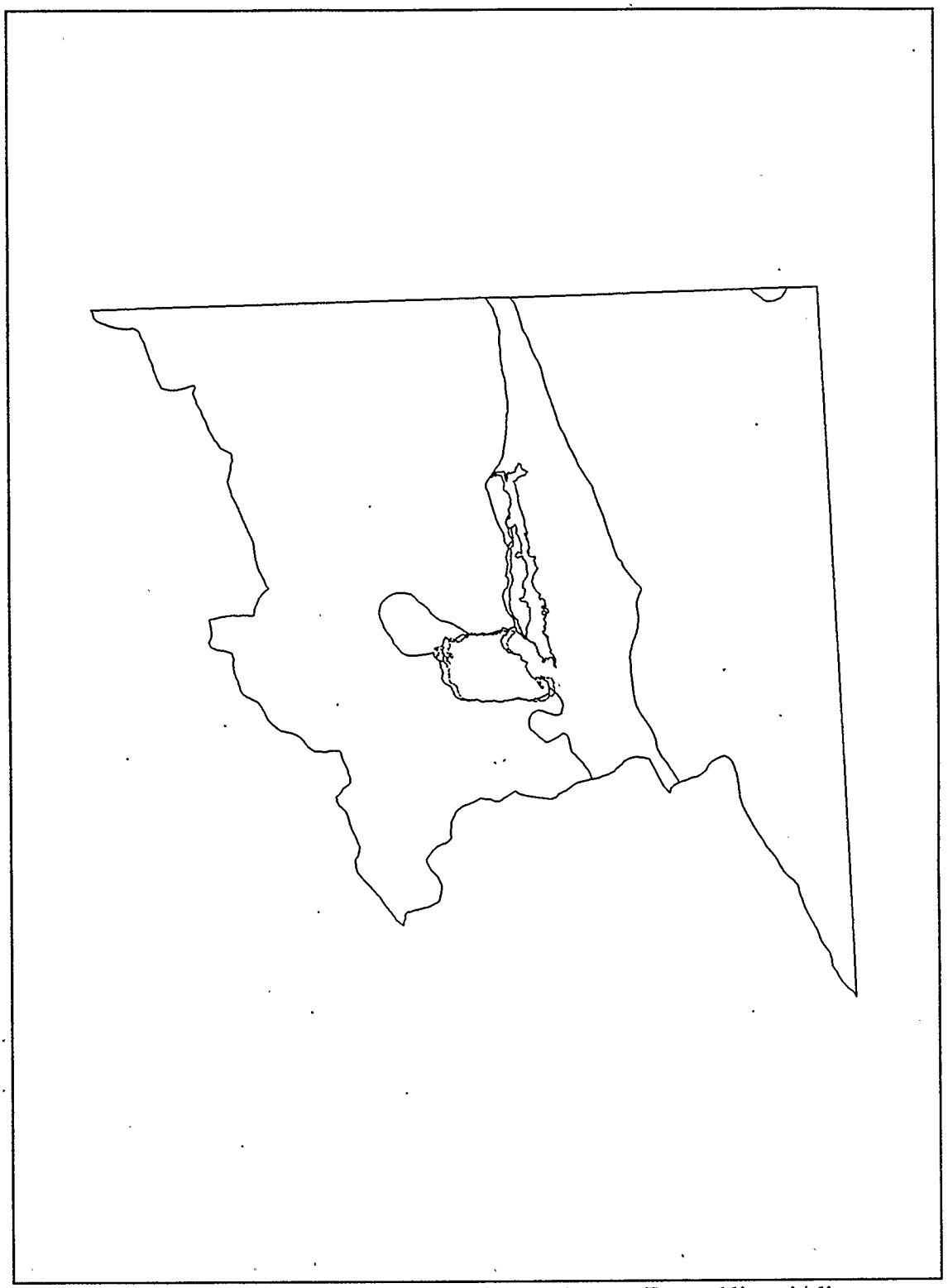

Figure 6.20 Automatic elimination of spurious polygons. Dotted lines indicate more accurate line work from 'lakes' which were eliminated. Enlarged from study area shown in Figure 5.1

eliminated. By supplying the operator with this data, the tedium and expense of editing can be reduced, and at the same time uncertainty reduction in the database can be achieved.

The approach taken involves performing a topological overlay of two coverages with a UNION operation, which produces an output coverage with all of the features of the input coverage. A human operator may be able, through visual inspection of an output coverage, to determine spatial relationships and problem areas. However, a topological overlay is required in order for UNCLE to be able to automatically determine feature level relationships. The 'soils' and 'lakes' coverages were used for this experiment, and produced the output coverage 'soilslakes'. Since other combinations of coverages also exhibited problems of inconsistent data when combined, they could also have been used. In ARC/INFO, the results from this overlay transformation are stored in an INFO table, which also records the polygon identifiers of the two input coverages. This provides a method of tracing the resulting polygons to their parent coverages. This INFO table is converted to an ASCII format, and the UNIX script INFO2PRO is then invoked. This program converts a specified file and feature type to a file of PROLOG clauses with predicates in the form 'cover_feature', where 'cover' is the coverage name and 'feature' refers to the INFO file type. The arity is set to the number of INFO items; the arguments for each clause contain the values of the original INFO records. INFO2PRO is written in a

general form, and takes command line parameters (coverage name and INFO file type), and therefore can be used for any coverage in the workspace. For example, the 'soilslakes' coverage 'pat' INFO file is converted to rules of the form

soilslakes_pat(recnum, area, perimeter, soilslakes#, soilslakes-id, soils#, soils-id, lakes#, lakes-id).

The file is then read into UNCLE and incorporated into the rule-base.

Rules are then added which define what will constitute an inconsistent polygon. This step is required for any coverage since the meaning of inconsistent will be different for different coverages. For the example chosen, the definition of inconsistency is based on the polygon identifiers. To determine which polygons are inconsistent, an examination of the logical consistency of these identifiers is performed. Referring to Figure 6.18, the lakes in the 'soils' coverage were assigned identifiers of zero. All other valid soils polygons were assigned unique integer values. In the 'lakes' coverage, the two lake polygons were assigned unique integer values, and all areas outside them - the envelope polygon - were given identifiers of zero.

If these two input layers were consistent with each other, all common feature boundary lines would be exactly the same, and the UNION overlay operation which produces 'soilslakes' should have exactly the same number of polygons as the 'soils' coverage, i.e. five. In addition, the combination of attributes for each polygon in the output coverage should have a non-zero value for the 'soils-id' field, and a zero value in the 'lakes-id' field. Any other combination of polygon identifiers present in the output coverage records would indicate an inconsistency. For example, a 'non-

zero/non-zero' (soils/lakes) pair would indicate that a soils polygon lay in a lake. This is an obvious logical inconsistency of the geographic phenomena. Such a polygon would obviously be spurious in this case. Other fields from the INFO file can potentially be used to determine logical inconsistencies. For example, attribute data entered for the 'soils', coverage and the names of lakes entered for 'lakes' could be used. A rule in UNCLE to define an inconsistency might be any polygon which has attributes 'perennial forage crops only, excess water' and 'Upper Kananaskis Lake'.

When the coverages were overlaid, the resulting 'soilslakes' coverage contained 45 polygons, indicating numerous inconsistencies and spurious polygons. Since there should only be five valid soil polygons, it would be necessary in this case for an operator to find 40 polygons and edit these -individually - out of the database. In order to intelligently do this, data on the accuracy of the coverages is desirable.

To facilitate the identification of these for the operator, rules were entered into the database which defined inconsistent polygons based on their attributes, in this case, the combination of their identifiers. By defining inconsistency by the presence of 'zero/zero', 'zero/non-zero', and 'non-zero/non-zero' pairs of identifiers in the 'soilslakes_pat' clauses, 34 polygons were identified. To guide an operator in selection of which lines of these polygons to eliminate, UNCLE also produces a lineage report and, using the positional accuracy measures obtainable through the PROLOG rule base, generates estimates for the linework (see Figure 6.21). It is assumed that an operator would desire this type of data in order to make an intelligent decision in editing. For example, the less accurate line work of the soils

data would be removed in this case.

```
?- show_inc.
the following polygons have been found to be inconsistent
[0,5,7,8,9,11,13,14,15,17,18,19,20,21,22,23,25,26,28,29,
30,31,32,34,35,36,37,38,39,40,41,42,43,44]

Yes

?- ppd(8,soilslakes).

polygon number 8

area 8.926559999999999e+02
perimeter 1.688790000000000e+02
parent polygons are
soils polygon number 7 polygon id 0
lakes polygon number 2 polygon id 2
soilslakes is derived from original coverage kvlakes
        kvlakes
                source : nts
                scale : 1 : 50000
                method : manual_digitizing
                operator : canny_mulaku
soilslakes is derived from original coverage soils82j
        soils82j
                source : cli
                scale : 1 : 250000
                method : manual_digitizing
                operator : ross_miller
estimated cf for coverage is [5.98090e+01,4.16403e+02]

Yes

?-
```

Figure 6.21 Report from UNCLE listing definite inconsistencies, with lineage and uncertainty measures included.

Although this method can quickly identify such user-defined inconsistencies, it is still necessary to process the data set further. All of the remaining 11 polygons had valid 'non-zero/zero' pairs of identifiers due to the nature of the linework. Nevertheless, since there should only be five valid soils polygons, some of these are obviously spurious. To identify these, a threshold value for area was determined and rules were added to search for polygons below that threshold. This approach

identified 6 out of 11 polygons. Of these one was in fact a valid polygon, and one of the five initially identified as valid was an artifact of the overlay process and was spurious. These exceptions demonstrate the difficulty of applying fully automated methods to establishing inter-layer consistency and removing spurious polygons, and the necessity for human intervention. The rule-base can be used to determine inconsistencies and print the results of the PROLOG queries, including lineage data, for the operator. The operator can then use these data in another window in ARC/INFO to display the polygons which are identified as inconsistent. Since the rule base can not automatically identify and delete the individual arcs which comprise the spurious polygons, the operator will likely need to interactively delete some of the arcs. This may require reasoning about such factors as the lineage of the coverages, and the application context. This sort of reasoning is not part of the prototype rule base, and consequently human intervention is required. The techniques employed in this example are applicable to other coverages in the data set, since they have been developed in a general fashion. However, as indicated, the user must define for each coverage the rules which govern the inconsistencies. The refinement of these methods can contribute to the reduction of uncertainty, in the form of inconsistencies, in a GIS database.

The rule base can also include topological and coordinate data for polygons and arcs, which is necessary to meet Requirement 1 (section 4.1). This is achieved by translating the coverage to a Digital Line Graph file, converting this to PROLOG facts, and then reading these into the rule base. The program DLG2LDB written for

database development purposes (Chapter 5) was copied to DLG2PRO and modified to produce an output file of PROLOG facts. These facts may define arc/node/polygon topology, polygon /arc topology, or arc coordinate data.

Arc/node/topology facts take the form

cover_arc(arc-id,npoints,from,to,left,right)

where 'cover' is the name of the original ARC/INFO coverage. The arguments are ordered the same as those for ODYSSEY CDB files.

Polygon/arc facts are in the form

cover_pal(polygon#, [arclist]).

where 'cover' is the name of the original ARC/INFO coverage. Polygon# is the internal polygon identifier used by ARC/INFO. It corresponds to the fifth argument in the 'cover_pat' files. The notation [arclist] refers to a PROLOG list notation, and is chosen because the number of arcs in a polygon is variable, and PROLOG permits variable length lists. In these lists, the arc-ids are listed in a clockwise fashion, as either positive or negative integer numbers separated by commas. The positive-negative convention defines the sense of a continuous linked lists of arcs. A negative arc-id simply indicates that the arc coordinate lists in the spatial files should be reversed when accessed, in order to ensure this continuity.

The arc coordinate files are in the form 'cover_xy', and take the form

cover_xy(arc-id,[coordinate_list]).

The naming convention is the same as the previous two facts. The arc-id is the same as that in the 'cover_arc' and the 'cover_pal' '[arclist]', except that these are always

positive. The '[coordinate_list]' is a list of consecutive x,y values.

The data in the 'cover_arc' facts are also accessible in the coverage AAT files, an INFO file. These also contain information on the internal arc number and the length of the arc. These can be converted to PROLOG facts by the INFO2PRO program, in the same manner as was done for the 'soilslakes' polygons. Because of these additional data it is a preferred method to derive these facts. The data in the internal PAL and ARC files are not accessible from within ARC/INFO by the user. However, by converting the coverage data to DLG format and then to facts, this is made available.

With these facts present in the rule base it is possible to completely recreate the topology of arcs and polygons, and utilize these data and their coordinates in conjunction with the 'cover_pat' files. Since the data also reside in a readable ASCII form, i.e. the DLG file, it can be utilized by external programs as well. However, this will require the modification of DLG to LDB to read the data into a form suitable for the external program.

6.6 Summary

A hybrid system of GIS software, operating system features and PROLOG rule base is developed and described. This system, UNCLE, facilitates the investigation of how various representations and methods of uncertainty can be used in conjunction with GIS software. Methods are developed to model lineage of GIS data, a type 4 uncertainty which has received little attention in the literature.

Through lineage, it is possible to define relations between GIS transformations and data sets and associate measures of uncertainties with these. This assists the study of error propagation in GIS software. Access to spatial and non-spatial data in ARC/INFO at coverage and feature level and procedures for incorporating these into the rule base are described. Methods for integrating external high-level languages for performing more complex computations are indicated.

Chapter 7

Conclusions and Recommendations

7.1 Conclusions

The main objective of this thesis was to identify and develop a computer environment which would support the investigation of uncertainty in Geographic Information Systems. Uncertainty is treated from within a communication paradigm which is concerned with all of the transformations GIS data may undergo from source to final product. It was determined that the topic of uncertainty is a very complex and important one in GIS, and the requirements for modelling and managing it include symbolic and numeric processing methods, which may require supporting different representations and management methods.

A hybrid computing environment was proposed and a prototype system developed which meets these requirements. Through this approach, and using primarily PROLOG rule-based techniques, it was demonstrated how the different requirements could be met. Experimentation was carried out to develop methods which would track the lineage of data from source to end product while incorporating uncertainty measures, and use this to reason about inconsistencies in GIS data.

The suite of tools which have been developed and described, in particular, the ability to access data from coverage level down to coordinate data, is a valuable contribution. Using these in external programs such as a PROLOG rule-base and in conjunction with GIS software and user programs builds a foundation for future

research and development in many areas.

7.2 Recommendations

Further research which utilizes UNCLE will depend on several factors. Most importantly, considerable work is required to develop the models of uncertainty for GIS data and transformations. Also, the ongoing debate in AI and expert systems research in uncertainty should be looked at in order to determine which methods of reasoning with uncertainty prove the most suitable for GIS use.

UNCLE currently works in a semi-automatic fashion. Through the use of ARC/INFO AMLs, and UNIX and C utilities, much of the file translations are transparent. Since UNCLE works concurrently with ARC/INFO in a separate process, the operator must still follow definite procedures to insure consistency between the rule base and the coverages. Some refinement of UNCLE is required in this area, to guide the user, via prompts and reminder messages, to the appropriate next action.

One area of particularly promising investigation is true UNIX inter-process communication between BIMprolog and ARC/INFO. This could essentially provide a real-time monitor of GIS transactions by the rule-base. Currently, it is possible to achieve inter-process communication with BIMprolog and C language programs.

The Department of Surveying Engineering should consider how to implement UNCLE on its existing computer facilities. Since the software was developed for ARC/INFO on the SUN workstations, it is reasonable to implement it on the

Departmental SUN equipment. However, since a version of a PROLOG language is required, additional software needs to be acquired. This can be done in one of two ways. BIMprolog can be acquired, in which case the existing prototype can be implemented directly. Alternatively, the existing Departmental licence for the VAX-based Quintus PROLOG software may be exchanged for a Quintus SUN-based version. This may require minor changes to the UNCLE code, however, since PROLOG code is relatively portable, this does not appear to be a major obstacle.

The Kananaskis data set is directly portable to the Departmental SUNs for use in ARC/INFO. However, in order to make this a better teaching and research data set some additional work is required. The cadastral polygon coverage is incomplete, so further data acquisition, probably digitizing, is required. The hydrology coverage, which contains rivers, streams, and smaller lakes, does not currently have toponymy associated with the features. This will require cross referencing some data from Mulaku (1987) with the ARC/INFO coverage. Likewise, the cadastral layer requires attribute data.

Additional data for the area exists which have not been incorporated in this research. For example, a digital Landsat image used by Paine (1987), and some elevation data in the Kananaskis Lakes region is known to be available in the Department in other formats. The GIS data set can be enhanced by converting these to ARC/INFO format. The conversion of the image data may be of particular merit to examine raster and vector integration in ARC/INFO or other GIS software.

Since the data set was developed on ODYSSEY, PAMAP and ARC/INFO some

interesting possibilities exist for comparative systems studies. It would be valuable to have other data as well. Most of the coverages are natural resource polygons, and additional linear data, particularly cultural features, should be acquired. Also, there is currently no survey control data in the data set. If the data set is extended to include these and possibly also extended geographically, it could be incorporated into the LIS component of the Departmental survey camp. These may provide some interesting fourth year projects, for example, for undergraduates.

References

Baldwin, J.F., (1986) Support Logic Programming, International Journal of Intelligent Systems, Vol. 1, pp. 73-104.

Barnett, J.A., (1981) Computational Methods for a Mathematical Theory of Evidence, Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, British Columbia, Canada, pp. 868-875.

Bédard Y., (1986a) A Study of the Nature of Data using a Communication-Based Conceptual Framework of Land Information Systems, unpublished Ph.D. dissertation, University of Maine, Orono.

Bédard, Y., (1986b) A Study of the Nature of Data using a Communication-Based Conceptual Framework of Land Information Systems, The Canadian Surveyor, Vol. 40, No. 4, Winter 1986, pp. 449-460.

Bédard, Y., (1987) Uncertainties in Land Information Systems Databases, Proceedings Auto-Carto 8, pp. 175-184, Baltimore, March 29-April 3.

BIM, (1990) Prolog by BIM, Version 3.0, BIM, Everberg, Belgium.

Blais, J.A.R., and Boulianne, M., (1988) Comparative Analysis of Information Measures for Digital Image Processing, International Archives of Photogrammetry and Remote Sensing, Vol. 27, Part B8, Commission III, Kyoto, pp. 34-44.

Blais, J.A.R. (1991) On Some Practical Applications of Information Theory in Geomatics, CISM Journal ACSGC, Canadian Institute of Surveying and Mapping, Ottawa, Volume 45, Number 2, pp. 239-247

Bolstad, P.V., Gessler P., and Lillesand, T.M., (1990) Positional uncertainty in manually digitized map data, International Journal of Geographical Information Systems, Volume 4, Number 4, October-December 1990, pp. 399-412.

Bratko, I., (1990) Prolog Programming for Artificial Intelligence, Second Edition, Addison Wesley.

Burrough, P.A., (1986) Principles of Geographical Information Systems for Land Resources Assessment, Oxford University Press.

Campbell, J., (1984) Introductory Cartography, Prentice-Hall Inc., Englewood Cliffs, New Jersey.

CCSM (Canadian Council on Surveying and Mapping), (1984) National Standards for the Exchange of Digital Topographic Data, Topographical Survey Division, Surveys and Mapping Branch, Energy, Mines and Resources Canada, Volume 1, p. 144.

Chrisman, N., (1982) A Theory of Cartographic Error and its Measurement in Digital Data Bases, Proceedings, Auto-Carto 5, pp. 159-168.

Chrisman, N., (1983) The Role of Quality Information in the long-term functioning of a Geographic Information System, Proceedings
of the Sixth International Symposium on Computer Assisted Cartography, Volume 1, pp. 308-312.

Cohen, P.R., (1985) Heuristic Reasoning with Uncertainty: An Artificial Intelligence Approach, Pitman Advanced Publishing Program, London, England.

Cowen, D., (1988) GIS versus CAD versus DBMS: What are the Differences?, Photogrammetric Engineering and Remote Sensing, Vol. 54, No. 11, November pp. 1551-1555.

Davis, J.C., (1986) Statistics and Data Analysis in Geology, Second Edition, John Wiley and Sons, Toronto.

Dubois, D. and Prade, H., (1987) A tentative comparison of numerical approximate reasoning methodologies, International Journal of Man-Machine Studies, Volume 27, pp. 717-728.

Fisher, P.F., (1989) Knowledge-based approaches to determining and correcting areas of unreliability in geographic databases, in Accuracy of Spatial Databases, Goodchild M. and Gopal, S., editors, Taylor and Francis.

Franklin, W.R., (1984) Cartographic errors symptomatic of underlying algebra problems, Proceedings of the International Symposium on Spatial Data Handling, Zurich, Switzerland, August 20-24, pp. 190-208.

Garvey, T., (1987) Evidential Reasoning for Geographic Evaluation for Helicopter Route Planning, IEEE Transactions On Geoscience and Remote Sensing, Vol. GE-25, No. 3, May, pp. 294-304.

Garvey, T., Lowrance, J.D., and Fischler, M.A., (1981) An Inference Technique for Integrating Knowledge from Disparate Sources, Proceedings of the 7th Joint Conference on Artificial Intelligence, American Association for Artificial Intelligence, Menlo Park, California, pp. 319-325, August

Goodchild, M. and Gopal, S., editors, (1989) Accuracy of Spatial Databases, Taylor and Francis.

Grady, R.K., (1988) Data Lineage In Geographic Information Systems (LIS/GIS), GIS/LIS '88, San Antonio, pp. 722-729.

Hamilton, A.C., and Williamson, I.P., (1985) A critique of the FIG Definition of 'Land Information System', Proceedings of FIG Commission 3, International Symposium on Land Information Systems, Edmonton, Alberta, October 1984, CIS, Ottawa, pp. 28-34.

Head, C.G., (1984) The Map as Natural Language, Cartographica, Vol. 21, No. 1, pp. 1-32.

Hinde, C.J., (1986) Fuzzy Prolog, International Journal of Man-Machine Studies, Volume 24, pp. 569-595.

Kitzmiller, C.T, and Kowalik, J.S. (1987) Coupling Symbolic and Numeric Computing in Knowledge-based Systems, AI Magazine, Summer 1987, pp. 85-90.

Klir, G., (1987) Where do we stand on measures of uncertainty, ambiguity, fuzziness, and the like ?, Fuzzy Sets and Systems, Vol. 24, pp. 141-160.

Lanter, D.P., (1990) Lineage in GIS: The Problem and a Solution, NCGIA Technical Paper 90-6, University of California, Santa Barbara, California.

Lanter, D.P., and Veregin, H., (1990) A Lineage Meta-Database Program for Propagating Error in Geographic Information Systems, Proceedings GIS/LIS '90, American Congress on Surveying and Mapping, Volume 1, pp. 144-153.

Lee, N.S., Grize, Y.L., Dehnad, K., (1987a) Quantitative Models for Reasoning under Uncertainty in Knowledge-Based Expert Systems, International Journal of Intelligent Systems, Vol. II, pp. 15-38.

Lee, T., Richards, J., and Swain, P., (1987b) Probabilistic and Evidential Approaches for Multisource Data Analysis, IEEE Transactions on Geoscience and Remote Sensing, Vol. GE-25, No. 3, May, pp. 283-292.

Leung, Y., (1987) On the Imprecision of Boundaries, Geographical Analysis, Vol. 19, No. 2, pp. 125-151.

Lodwick, G.D., (1986) Parcel-based Land Information Systems, Publication No. 10010, Department of Surveying Engineering, University of Calgary, Calgary, Alberta.

Lodwick, G.D., Paine, S., Mepham, M. and Colijn, A., (1986), A comprehensive LRIS of the Kananaskis Valley using Landsat data, Proceedings of the Seventh International Symposium on Remote Sensing for Resources Development and Environmental Management, International Society for Photogrammetry and Remote Sensing, Commission VII, Enschede, The Netherlands, 25-29 August, pp. 927-932.

Meadow, C., (1973) The Analysis of Information Systems, Melville Publishing Company.

Mepham, M., (1988) Integration of Cadastral Survey Plans Into a Computerized Land Information System, unpublished Ph.D. thesis, University of Calgary, Department of Surveying Engineering, Publication No. 20029.

Miller, R.G., Lodwick, G.D. and Allen, D.E., (1989) Evaluating Uncertainty in Operational GISs, Proceedings, AM/FM International Conference XII, New Orleans, La., April 10-13 pp. 323-336.

Mulaku, G.C., (1987) Map Data Digitizing, Editing and Automatic Hydrological Network Reconstruction, unpublished Master's Thesis, University of Calgary, Department of Surveying Engineering, Publication No. 20021.

NCDCDS (National Committee for Digital Cartographic Data Standards), (1988) The Proposed Standard for Digital Cartographic Data, The American Cartographer, Vol. 15, No. 1, pp. 9-140.

NCGIA, (1989) The research plan of the National Center for Geographic Information and Analysis, International Journal of Geographic Information Systems, Volume 3, pp. 117-136.

Newcomer, J. and Szajgin, J., (1984) Accumulation of Thematic Map Errors in Digital Overlay Analysis, The American Cartographer, Vol. 11, No. 1, pp. 58-62.

ODYSSEY (1982), ODYSSEY User's Reference Manuals, Version 1.0, Harvard University Graduate School of Design, Laboratory for Computer Graphics and Spatial Analysis, Cambridge, Massachusetts.

Openshaw, S., (1989) Learning to live with errors in spatial databases, in Accuracy of Spatial Databases, Goodchild M. and Gopal, S., editors, Taylor and Francis, pp. 263-276.

Paine, S., (1983) Position-Based Surface-Cover Mapping in the Kananaskis Valley Using Digital Landsat Data, unpublished Master's Thesis, University of Calgary, Department of Surveying Engineering, Publication No. 20002.

Paine, S., (1987) Information Extraction from Digital Landsat Imagery for Integration into an LRIS, unpublished Ph.D. Thesis, University of Calgary, Department of Surveying Engineering, Publication No. 20026.

Petersohn, C., and Vonderohe, A., (1982) Site Specific Accuracy of Digitized Property Maps, Proceedings of Auto-Carto 5, pp. 607-619, Crystal City, Virginia, 22-28 August.

Peucker, T. and Chrisman, N., (1975) Cartographic Data Structures, American Cartographer, Vol. 2, No. 1, pp. 55-69.

Robinove, C.J., (1981) The Logic of Multispectral Classification and Mapping of Land, Remote Sensing of Environment, Volume 11, pp. 231-244.

Robinson, A., Sale, R., Morrison J., and Muehrcke, P., (1984) Elements of Cartography, Fifth Edition, John Wiley and Sons, Toronto.

Robinson, V. and Frank, A., (1985) About Differing Kinds of Uncertainty in Collections of Spatial Data, Proceedings of Auto-Carto 7, pp. 440-449, Washington, 11-14 March.

Robinson, V., and Frank, A., (1987) Expert Systems for Geographic Information Systems, Photogrammetric Engineering And Remote Sensing, Vol. 52, No. 10, pp. 1435-1441.

Robinson, V. and Strahler, A., (1984) Issues in Designing Geographic Information Systems Under Conditions of Inexactness, Proceedings, Tenth International Symposium on Machine Processing of Remotely Sensed Data, Purdue University, pp. 198-204.

Robinson, V., and Thongs, D., (1985) Fuzzy Set Theory Applied to the Mixed Pixel Problem of Multispectral Landcover Databases, Technical Proceedings of the Workshop on Geographic Information Systems in Government, U.S. Army Engineer Topographic Laboratories, Volume 2, pp. 871-885, Springfield, Virginia, 10-13 December.

Shannon, C., and Weaver, W., (1959) The Mathematical Theory of Communication, University of Illinois Press, Urbana, Illinois.

Shafer, G., (1976) The Mathematical Theory of Evidence, Princeton University Press.

Shine, J. A., (1985) Bayesian, Evidence, Fuzzy: Which Theory Works Best When Reasoning With Uncertain Knowledge ?, Technical Papers of the 51st Annual Meeting of the American Society of Photogrammetry, Washington, D.C., March 1985, ASPRS, Falls Church, Virginia, Vol. 2, pp. 676-679.

Sterling, L. and Shapiro, E., (1986) The Art of Prolog - Advanced Programming Techniques, The MIT Press, Cambridge, Massachusetts.

Stoms, D., (1987) Reasoning with Uncertainty in Intelligent Geographic Information Systems, Second Annual International Conference, Exhibits and Workshop on Geographic Information Systems, San Francisco, 1987, pp. 693-700.

White, D., (1978), A Design for Polygon Overlay, First International Advanced Study Symposium on Topological Data Structures For Geographic Information Systems, Harvard Papers on Geographic Information Systems, Dutton, G. editor, Harvard University Graduate School of Design, Laboratory for Computer Graphics and Spatial Analysis.

Yager, R., (1986a) Toward a General Theory of Reasoning with Uncertainty. I: Nonspecificity and Fuzziness, International Journal of Intelligent Systems, Vol. 1, pp. 45-67.

Yager, R., (1986b) Toward a General Theory of Reasoning with Uncertainty. Part II: Probability, International Journal of Man-Machine Studies, 25, 613-631.

Zadeh, L., (1965) Fuzzy Sets, Information and Control, Vol. 8, pp. 338-353.

Zhang, G., (1989) Modelling Consistent Transactions Within Automated Cadastral Land Information Systems, unpublished Master's Thesis, University of Calgary, Department of Surveying Engineering, Publication No. 20030.