Design and Implementation of an Object-Oriented Space-Time GIS Data Model

A Thesis Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Ziliang Zhao
August 2011

Copyright © 2011 by Ziliang Zhao
All rights reserved.

ACKNOWLEDGEMENTS

I am grateful to my advisor, Dr. Shih-Lung Shaw, for his continuous and patient guidance and support on my study. The research idea of this research came originally from Dr. Shaw. I further explored and implemented this idea in the past 2 years through numerous meetings with Dr. Shaw. He also helped me a lot on thesis writing, which turned me from a pure technician toward a serious researcher. Although my work is still not perfect, I am already on the right track.

Many thanks go to Dr. Bruce Ralston and Dr. Liem Tran. They always offered me help and suggestions. The knowledge I gained from Dr. Ralston and Dr. Tran was beneficial to the model design and prototype system development in my thesis research.

I really appreciate my parents and grandparents in China. Although they could not help me on my research, their warm and valuable encouragements support me to reach this point.

ABSTRACT

Geographic data are closely related to both spatial and temporal domains. Geographic information systems (GIS) can capture, manage, analyze, and display spatial data. However, they are not suitable for handling temporal data. Rapid developments of data collection and location-aware technologies stimulate the interests of obtaining useful information from the historical data. Researchers have been working to build various spatio-temporal data models to support spatio-temporal query. Nevertheless, the existing models exhibit weaknesses in various aspects. For instance, the snapshot model is plagued with data redundancy and the event-based spatio-temporal data model (ESTDM) is limited to raster dataset. This study reviews existing spatio-temporal data models in order to design an object-oriented space-time GIS data model that makes additional contributions to processing spatio-temporal data. A binary large object (BLOB) data type, labeled Space-Time BLOB, is added to ArcGIS geodatabase data model to store instantiated space-time objects. A Space-Time BLOB is associated with an array that contains the spatial and temporal information for an object at different time points and time intervals. This study also implements a space-time GIS prototype system, along with a set of spatio-temporal query functions, based on the proposed space-time GIS data model.

TABLE OF CONTENTS

Chapter 1 Introduction 1
1.1 Research Background 1
1.1.1 Time as an essential property 1
1.1.2 Relationships between space and time 2
1.1.3 The need to address spatio-temporal queries related to both space and time 2
1.1.4 The limited capability of the current space-time GIS data model in addressing spatio-temporal queries 3
1.2 Research Objectives and Research Questions 4
1.2.1 Research objectives 4
1.2.2 Research questions 4
1.3 Organization of the Thesis 5
Chapter 2 Literature Review 6
2.1 Spatio-temporal Query 6
2.2 Spatio-temporal Data Models 8
2.2.1 The snapshot model 10
2.2.2 The simple time-stamping model 11
2.2.3 The space-time composite model 12
2.2.4 The amendment model 14
2.2.5 The grid model 14
2.2.6 Event-based spatio-temporal data model (ESTDM) 15
2.2.7 Three-domain model 16
2.2.8 Object-oriented model 17
2.2.9 The extended dynamic GIS model (EDGIS) 18
2.2.10 ESRI parcel model 19
2.2.11 The TimeObject in the national cooperative highway research program (NCHRP) and other types of time 21
2.3 Spatio-temporal Visualization 23
2.4 Moving Objects Database 24
Chapter 3 An Object-Oriented Space-Time GIS Data Model 27
3.1 Relevant Conceptualization of Space and Time 27
3.1.1 The prevalence of the snapshot model and its limitation 27
3.1.2 Object-oriented view of space and time 28
3.2 Design of the Object-oriented Space-Time Class 29
3.3 Creation of Space-Time Object 33
3.3.1 Creation of Space-Time BLOB for snapshots with entity identity 33
3.3.2 Incorporation of new data into the Space-Time BLOB of object with entity identity 37
3.3.3 Creation of Space-Time BLOB for snapshots without entity identity 39
3.3.4 Incorporation of new data into the Space-Time BLOB of object without entity identity 41
3.4 Storage of the Space-Time BLOB 42
Chapter 4 A Space-Time GIS Prototype System 44
4.1 The Prototype System and the Implementation Platform 44

4.2 System Overview 47
4.3 Tools for Space-Time BLOB Creation in Geodatabase 48
4.3.1 Data Preparation before processing 48
4.3.2 Tools to Create Space-Time BLOB for snapshots with entity identity 49
4.3.3 Tools to Create Space-Time BLOB for snapshots without entity identity 52
4.4 Functions of Spatial-temporal Query 57
4.4.1 Interface design of Spatio-temporal query 57
4.4.2 What -> When + Where 58
4.4.3 When -> What + Where 60
4.4.4 Where -> What + When 62
4.4.5 What + When -> Where 63
4.4.6 What + Where -> When 63
4.4.7 Where + When -> What 65
Chapter 5 Conclusions 67
5.1 Summary of the Study 67
5.2 Future Research Directions 68
5.2.1 Incorporating other forms of time 68
5.2.2 System implementation for point, polyline and raster 69
5.2.3 Incorporation with analytical time-geographic framework 70
5.2.4 Spatio-temporal analysis 70
5.2.5 Processing power on very large dataset 70
REFERENCES 72
VITA 76

LIST OF TABLES

Table Page

Table 2.1 Sinton's measurement framework (Sinton 1978)..... 8
Table 2.2 Classification of spatio-temporal data model 9
Table 3.1 The generated table of greatest common units 40

LIST OF FIGURES

Figure 2.1 Temporal topology 7
Figure 2.2 Snapshot model 11
Figure 2.3 The time tab in the layer properties dialog in ArcMap 10 12
Figure 2.4 The space-time composite model 13
Figure 2.5 The amendment model 15
Figure 2.6 The grid model 15
Figure 2.7 Event-based spatio-temporal data model 16
Figure 2.8 Object-oriented model 18
Figure 2.9 The data structure of EDGIS 19
Figure 2.10 ESRI parcel model 20
Figure 2.11 Time object in NCHRP 20-27(3) 22
Figure 2.12 Different types and structures of time 22
Figure 2.13. Four representation methods in ESTAT 24
Figure 3.1 The UML diagram of the proposed space-time GIS design 30
Figure 3.2 The flowchart of creating Space-Time BLOBs for snapshots with entity identity 34
Figure 3.3 The flowchart of incorporating new data into Space-Time BLOBs with entity identity 38
Figure 3.4 An example of generating greatest common units 40
Figure 3.5 The process of serialization 43
Figure 4.1 Historical Chinese province maps 45
Figure 4.2 Historical land use parcels on UT campus 46
Figure 4.3 The interface of the prototype system 48
Figure 4.4 The organization of the space-time Geodatabase 50
Figure 4.5 The tab of Space-Time BLOB creation 50
Figure 4.6 Beijing's Space-Time Point in 1953 51
Figure 4.7 Beijing's Space-Time Interval from 1964 to 2000 51
Figure 4.8 Chongqing's Space-Time Point in 2000 53
Figure 4.9 Fujian's Space-Time Interval from 1953 to 2000 53
Figure 4.10 Heilongjiang's Space-Time Point in 1953 53
Figure 4.11 Heilongjiang's Space-Time Point in 1964 53
Figure 4.12 Heilongjiang's Space-Time Point in 1982 53
Figure 4.13 Heilongjiang's Space-Time Point in 1990 53
Figure 4.14 Heilongjiang's Space-Time Point in 2000 54
Figure 4.15 The generated greatest common units 54
Figure 4.16 An example of a greatest common unit with a Space-Time Interval from 1998 to 2008 55
Figure 4.17 Another example of a greatest common unit with a Space-Time Interval from 1998 to 2003 and a Space-Time Point in 2008 56
Figure 4.18 An example of the lineage of a greatest common unit 56
Figure 4.19 Another example of the lineage of a greatest common unit 58
Figure 4.20 Interface design of spatio-temporal query 58
Figure 4.21 Query type selection 59

Figure 4.22 The settings of spatio-temporal query “What -> When + Where”..... 60
Figure 4.23 The result of spatio-temporal query “What -> When + Where” 60
Figure 4.24 The settings of spatio-temporal query “What -> When + Where”..... 60
Figure 4.25 The result of spatio-temporal query “What -> When + Where” 61
Figure 4.26 The settings of spatio-temporal query “When -> What + Where”..... 62
Figure 4.27 The result of spatio-temporal query “When -> What + Where” 62
Figure 4.28 The settings of spatio-temporal query “Where -> What + Where”..... 63
Figure 4.29 The result of spatio-temporal query “Where -> What + When” 64
Figure 4.30 The settings of spatio-temporal query “What + When -> Where”..... 64
Figure 4.31 The result of spatio-temporal query “What + When -> Where” 64
Figure 4.32 The settings of spatio-temporal query “What + Where -> When”..... 65
Figure 4.33 The result of spatio-temporal query “What + Where -> When” 66
Figure 4.34 The settings of spatio-temporal query “Where + When -> What”..... 66
Figure 4.35 The result of spatio-temporal query “Where + When -> What” 66

CHAPTER 1 INTRODUCTION

Both space and time are key properties of geographic data. Geographic information systems (GIS) handle the spatial domain of the geographic data very well in data management, data analysis, and data visualization. However, it is well known that GIS have very limited capabilities in answering the question with temporal domain, such as "how does the boundary of Beijing change over time." The goal of this study is to design a space-time GIS model to organize the spatio-temporal data in a way so that certain kinds of spatio-temporal queries could be easily addressed. Chapter 1 presents research background, research questions, and objectives. The organization of the thesis is outlined at the end this chapter.

1.1 Research Background

This section demonstrates the background of the research. The importance of time and its relationship with space is clarified first, followed by the need to answer different types of questions related to both space and time based upon spatio-temporal dataset. How current GIS can do this and its limitations are also discussed.

1.1.1 Time as an essential property

Everything in the universe is associated with time, whether that "thing" is tangible or intangible. Tangibles are substances that we can see, touch, and feel; they exist everywhere in the world. Human beings are a typical example of this category in that they possess substance, and experiences like birth, graduation, marriage, death can all be stamped with time. One may not remember the exact times of these events, but they do exist. Land use parcel is another example of tangibles that change over time. When people enter the parcel transactions in the database, for example, parcel owner change, the subdividing of a piece of property, the time when these transaction take place must be recorded.

Intangibles do not physically exist in the real world, but they are closely related to our lives, usually referring to activities or events. An activity is a process, such as a colloquium in the Geography Department, which is associated with a start time and an end time. An event is an occurrence or an outcome that has significance. We refer to an event when we encounter an old friend on the way to class. An event can sometimes be the start or end of an activity, and it sometimes triggers another, subsequent, activity. The submission of an order to Amazon is an event and triggers the packaging and shipment of the ordered items.

1.1.2 Relationships between space and time

In fact, the data we capture in the real world are closely related to both spatial and temporal domains because things change in space over time (Peuquet 2002). Most living things, such as human beings, keep changing location in order to conduct different activities to meet their needs. The position at one time point is probably not the same as the location at the next. Non-living things may move through time, or may not. The former involves vehicles, satellites, and so forth. The latter includes buildings, land parcels, and so on. In this case, even if the position could be consistent, properties such as physical existence and attributes change through time: a new building could be dismantled after fifty years; a land parcel's appraisal might change frequently (Arctur and Zeiler 2004).

However, a large number of GIS students and experts pay attention only to the spatial representation without realizing the importance of the temporal information. When using a digital map for analysis in the GIS lab class, very few students would ask at what time the features in the map were captured and digitized. The majority of students take for granted that the data they use represent the "correct" state of the world, while in fact this assumption might not be correct even if the data is the latest released version. Thus, the conclusions drawn from the spatial analysis might be inaccurate.

In short, space and time are both key properties possessed by each object in our world. On the one hand, if we ignore space, we lose the most significant geographic information, that is, location. On the other hand, if we ignore time, we cannot understand when a particular state exists or when a spatial change takes place, making location information less meaningful.

1.1.3 The need to address spatio-temporal queries related to both space and time

Many people are very interested in the performing queries in order to find historical information. They expect GIS to provide the answer not only in spatial, but also in temporal domains. Consider the following scenarios.

A historian uses GIS to study the historical change of Chinese province boundaries. He has the static maps of Chinese province boundaries in each year during 1950 to 2000, suggesting about 50 layers in total. To study the stability of Chinese province, the historian is curious about how many provinces had boundary changes between 1950 and 1960, 1960 and 1970, and so forth. He is also interested in how Rehe Province changed in 1950s, what adjacent provinces merged its land after Rehe Province was removed. Also, the historian wants to know how Beijing's boundary changed over time and what adjacent provinces were influenced.

In the past 20 years, Knox County Metropolitan Planning Commission (MPC) has been maintaining the land use data in Knox County. A land use analyst obtains this dataset in order to study the land use change in Knox County in the past 20 years. He wants to know how many land use parcels changed from residential to commercial between 2000

and 2005. Among these parcels, which changed back to commercial land parcel again in 2008 and were these land use type changes associated with land owner changes?

These are two typical examples which illustrate that the information many people are interested in are related to the temporal domain. In fact, such spatio-temporal query capability could benefit many other areas, such as human migration, environmental evolution, and so forth.

In recent years, developments in data collection technologies have produced huge amounts of spatio-temporal datasets. Compared with previous decades, people now have more and more spatio-temporal data, which enable researchers to dig out hidden information accumulated over time to unveil the changes, patterns, and processes. Therefore, the needs for powerful spatio-temporal query functions are even stronger.

1.1.4 The limited capability of the current space-time GIS data model in addressing spatio-temporal queries

GIS can capture, manage, analyze, and display spatial data. However, they are not well suited for handling data with temporal domain. The most prevalent and widely applied is the snapshot data model (Armstrong 1988). This model is simple and easy to use. To address the foregoing mentioned spatio-temporal queries, people often use snapshot model in commercial GIS packages, such as ArcGIS. Snapshots which represent the states of the study area at different times are stacked as layers. People have to extract relevant information layer by layer. After then, they have to compare the objects from consecutive time points to determine if changes occur. Since snapshot model does not explicitly maintain the information about how things change over time, people have to do it manually. Imagine if one has 10,000 layers, answering the questions, such as "how did object A change" leads to tremendous workload. Even if a script could be developed to automate this process, selecting and comparing objects layer by layer is inefficient.

From the data storage perspective, snapshot model has data redundancy problem. In the land use scenario, each land parcel has to be recorded again and again, even if no change occurs on this parcel during that time. Therefore, the data redundancy could be huge when people try to use snapshot model to record relatively stable area every short period.

Researchers have made considerable effort to design new space-time GIS data models to replace the snapshot model, such as event-based, object-oriented, and three-domain models. Nevertheless, existing models exhibit weaknesses, such as data redundancy, implementation difficulties, and so forth. Moreover, in most studies, the proposed space-time GIS models only deal with hypothetical data. Their ability to handle large real-world historical dataset is questionable. In summary, current space-time GIS data models are far from satisfactory. Snapshot model is still the dominant one.

1.2 Research Objectives and Research Questions

A series of research challenges are involved in this study. In this section three major objectives of this study are listed. To achieve these objectives, five research questions are also identified and discussed.

1.2.1 Research objectives

The proposed study aims to pursue three main research objectives: (1) design a space-time GIS model to store the spatio-temporal data to facilitate the capability and performance on certain spatio-temporal queries, such as how does an object change over time. This data model would be object-oriented which explicitly records the spatio-temporal change of the object. Using this model, an object does not need to be recorded if no change occurs; (2) to design a workflow to organize historical data which have been accumulated for some time, as well as a workflow to incorporate new data. Object with or without entity identity are both considered; (3) to implement a prototype system based on the designed data model, which includes the automatic historical data processing (i.e., automatically loads objects from snapshots into Geodatabase), the storage of the space-time objects, and the establishment of spatio-temporal query functions.

1.2.2 Research questions

The following research questions are addressed to achieve the above objectives:

1. Spatio-temporal queries: Peuquet (Peuquet 1994) suggested three types of spatio-temporal queries: "when + where → what," "when + what → where," and "where + what → when." In addition, the following three types of spatio-temporal queries are possible: (a) "when → what + where," (b) "what → when + where," and (c) "where → what + when." In these queries, only one of the components is controlled so more results could fit the requirement. For instance, when conducting the "when → what + where" query on travel diary data, the system may generate a list indicating all individuals (what) and their corresponding locations (where) at the given time. A sound and efficient space-time GIS data model enables powerful spatio-temporal query functions. What information should be maintained in the proposed space-time GIS data model so that the change of objects over time could be tracked? How can a proper user interface be designed to facilitate spatio-temporal queries?

2. Time representation in GIS: The traditional snapshot model records the objects at a given time point. To reduce data redundancy, time interval is introduced. How can we integrate time interval and use it in conjunction with time point to represent temporal information?

3. Management of different types of historical data: There are two kinds of historical spatio-temporal data. One is with entity identity, such as the historical Chinese province maps. The other is without entity identity, such as the land use data.

The key difference is that an object with entity identity is associated with a persistent ID that distinguishes one object from another, while the object without entity identity simply occupies some space in the real world within which the attributes are the same. How can we preserve the spatio-temporal information for both types of object? What do the differences of object with/without entity identity imply in the space-time GIS data model design? Can we use the same mechanism for both situations?

4. Storage and access of space-time objects: After creating a space-time object using the proposed data model, it must be stored for future access and analysis. How can this space-time object be transferred to a binary file and stored in the Geodatabase? How can the binary file be resurrected back into a space-time object when needed?

1.3 Organization of the Thesis

This thesis is organized into five chapters. Chapter 2 reviews the literature covering what have been done related to this study, including spatio-temporal data models, spatio-temporal query and other related topics. In Chapter 3, the limitation of the dominant space-time GIS data model, snapshot model, is discussed, followed by the idea of object-oriented view of space and time. This chapter also illustrates the design of the proposed object-oriented space-time GIS data model, as well as the workflows to organize and maintain the spatio-temporal data based upon the suggested model. Chapter 4 demonstrates the implemented prototype system: organizing historical data using the proposed object-oriented space-time GIS data model, accessing and interpreting the spatio-temporal information, conducting spatio-temporal queries and understanding the result. Chapter 5 summarizes the achievements and contributions of this study. This study serves as an important part of ongoing development of a comprehensive space-time GIS framework led by my advisor, Dr. Shih-Lung Shaw.

CHAPTER 2 LITERATURE REVIEW

A GIS serves as a powerful platform for dealing with spatial data. However, it lacks the ability to handle temporal information. Goodchild (1992) posed several challenges that constituted important research areas in geographic information science. Among these challenges is the modeling of time-dependent data. Some research has actively focused on this area, and many creative approaches have been proposed during the past two decades.

It should be noted that there is no perfect data model. Each spaito-temporal data model exhibits strengths when answering certain types of spatio-temporal query, while for other types of queries it might be inefficient. This chapter starts from spatio-temporal query, followed by the review of several data models which are used to support the query. Spatio-temporal visualization which could be used to better demonstrate the result is also discussed in this chapter.

2.1 Spatio-temporal Query

Spatio-temporal query dig out information from the spatio-temporal dataset and provide useful information to the user. The capability of spatio-temporal query varies in terms of the underlying spatio-temporal data model. This section discusses several topics relevant to spatio-temporal query.

The most important feature that distinguishes a spatio-temporal query from traditional spatial query in current commercial GIS package is that a spatio-temporal query contains a temporal component, which is either specified in the condition or reflected from the result. When constructing the spatial component, people use spatial topology to specify the location with reference to a known position, such as "within 100 meters of I-40," "inside the Knox County." This is very similar to specifying the temporal component. Allen (1984) proposed a temporal topology, including seven temporal relations: *before, equal, meets, overlaps, during, starts, and ends* (Figure 2.1). These seven scenarios are used to describe the conditions of temporal information when performing queries. In this thesis research, the prototype system is able to select all objects whose times of existence "*overlap*" or "*during*" the specified start and end times. In overlap relationship, the object with start time or end time of the existence between the specified time period would be selected, while the object with start time and end time both between the specified period would be selected in the "*during*" relationship. The mechanism of spatio-temporal query with temporal relationships other than "overlap" and "during" can be implemented by comparing the start time and end time that an object exists with the specified time period in different ways.

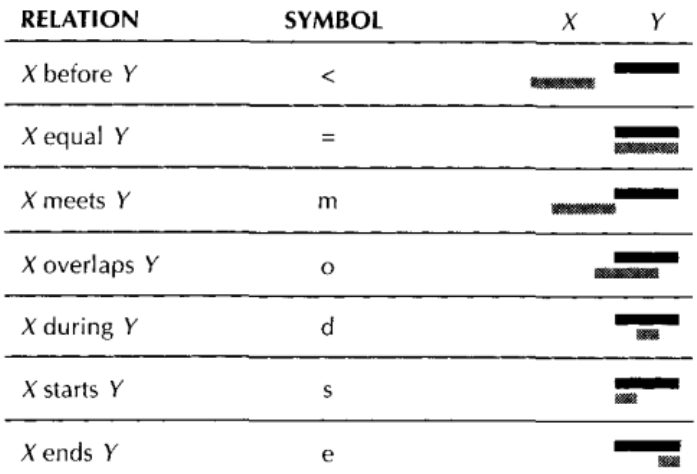


Figure 2.1 Temporal topology

(Allen 1984)

Spatio-temporal queries are composed in different forms. Peuquet (1994) classified data types into three categories: where, when, and what. Accordingly, any spatio-temporal query can be categorized into one of the following types of questions:

(1) when + where → what
(2) when + what → where
(3) where + what → when

This thesis research extends this classification by incorporating the following three types of spatio-temporal queries:

(1) what → when + where
(2) when → what + where
(3) where → what + when

Sinton (1978) proposed a framework for spatio-temporal query. In Sinton's framework, location, time, and attribute are treated as "fixed," "controlled," and "measured" components, respectively (Table 2.1). Six types of spatio-temporal queries were suggested in terms of these three components. Shaw and Xin (2003) used *Sinton's framework* to study land use and transportation interactions. Several spatio-temporal queries have been proposed in Shaw and Xin's article, such as "Which street segments that are adjacent to commercial land use in the study area had a traffic volume/capacity ratio greater than 1 in 1998?" This is a "when + what → where" question and the "during" relationship is involved. Moreover, in the query given above, time is fixed and the attribute is controlled while the location is measured. In their transportation demand modeling study, Wang and Cheng (2001) organized queries into four groups—time based, person based, activity based, and location based—which somewhat resemble Peuquet's classification.

Table 2.1 Sinton's measurement framework (Sinton 1978)

	Fixed component	Controlled component	Measured component
Scenario 1	Location	Time	Attribute
Scenario 2	Location	Attribute	Time
Scenario 3	Time	Location	Attribute
Scenario 4	Time	Attribute	Location
Scenario 5	Attribute	Location	Time
Scenario 6	Attribute	Time	Location

Temporal logic, which is used to describe any system of rules and symbolism for representing, and reasoning about, propositions qualified in terms of time, serves as a flexible temporal notation to describe the process (Hodkinson and Reynolds 2006). *Temporal logic* also has a series of temporal operators to represent the temporal relationship such as until, next, future, and so on.

To accommodate the needs of spatio-temporal query, *structured query language* (*SQL*) has been extended to *TSQL*, which incorporates the temporal domain (Snodgrass 1995). Aufaure-Portier and Bonhomme (1999) introduced a visual language denoted as *Lvis* to query spatial data. Instead of generating an *SQL* sentence, predefined icons modeling spatial objects and operators are applied to build the query. In addition, *Lvis* is extended to incorporate data with temporal information. New visual metaphors were proposed to express spatial and temporal criteria. These visual queries are implemented by translation into an *SQL* language with spatio-temporal extensions. In this thesis research, an interactive approach is adopted to conduct spatio-temporal query. The list of query parameters is populated with all possible values for the user to select the desired one. The prototype can return the query result based upon the SQL sentence which is automatically built by the system.

2.2 Spatio-temporal Data Models

A spatial data model abstracts the objects in the real world into digital representation in the computers. However, only the information at a specific moment is recorded. The states of objects at different times are unknown. Thus, spatio-temporal query cannot be supported. On the contrary, spatio-temporal data model records the objects through time.

Current spatio-temporal data models all exhibit strengths and weaknesses. The characteristics of these data model lead to varied performances on different spatio-temporal queries. These section reviews 10 most representative spatio-temporal data models by classifying them into 4 groups: object-oriented and event-based, object-oriented and non-event-based, class-oriented and event-based, and class-oriented and non-event-based. Each spatio-temporal data model belongs to one or more classes in terms of its characteristics (Table 2.2). Before discussing these 10 spatio-temporal data models in detail, it is important to understand the differences between object-oriented and class-oriented, and between event-based and non-event-based.

Table 2.2 Classification of spatio-temporal data model

	Event-based data model	Non-event-based data model
Object-oriented data model	Space-time composite model Amendment model Grid model Object-oriented model	Space-time composite model Amendment model Grid model
Class-oriented data model	Snapshot model Simple time-stamping model Space-time composite model Event-based spatio-temporal data model Three-domain model Extended dynamic GIS model ESRI parcel model	Snapshot model Space-time composite model Extended dynamic GIS model

Object-oriented vs. Class-oriented
Event-based model and non-event-based model are two different approaches regarding how the spatio-temporal information of objects would be stored.

Object-oriented data model focuses on each individual object. How this object changes through time is recorded encapsulated with the object, rather than being mixed with the data of other objects. As a result, the spatio-temporal information of the target can be easily extracted by accessing this object. The spatio-temporal query with "what" component in the condition, such as "how does object A change between 2000 and 2010", can be addressed very efficiently. However, it will not be that efficient if "what" is not a component in the condition. For example, people may be interested in finding all buildings that exists in 1998 (when → what + where). To handle this query, each building object has to be accessed. Its spatio-temporal information encapsulated within the object will then be analyzed.

Class-oriented data model aggregates all spatial objects and manages them in a single feature class or relational table. An object at different times would have different records, each of which represents its state at a time period. It is not as efficient as object-oriented approach when performing query to track changes on one or more objects. On the contrary, the class-oriented approach is convenient when "what" is not a component that needs to be specified, since the spatio-temporal information of all objects are within one table. For example, to find all provinces with population larger than one million, cycling through the attribute table is convenient. However, if object-oriented approach is adopted, each object should be accessed and then its population property would be analyzed.

Event-based vs. Non-event-based
Event-based model and non-event-based model are two different approaches regarding when the area under study would be recorded.

An event refers to a change in geometry or attribute of an object. An event could trigger a new record for the changed object (in object-oriented approach) or new records for all objects in the study area (in class-oriented approach). An event-based model makes sure that the exact time when changes occur could be stored in the database. Nevertheless, data volume could be tremendous if such event occurs too frequently.

Non-event-based model does not record the study area when events occur. Instead, it could be based on regular intervals, or even random time points. The weakness of this approach is obvious: the event times when the states of objects change are unknown.

The remainder of this section reviews 10 spatio-temporal data models with reference to the foregoing mentioned classification.

2.2.1 The snapshot model

The *snapshot model* is one of the simplest spatio-temporal data models (Armstrong 1988). With this model, spatio-temporal data consist of a series of snapshots (Figure 2.2). A snapshot records the state of a certain area under study at a specific time point.

On the one hand, the snapshot model could be event-based (i.e., a snapshot is taken when change occurs) or non-event-based (e.g., every Monday). On the other hand, the snapshot model is class-oriented. Each object that exists at the same time point would be recorded.

The *snapshot model* is the most widely used spatio-temporal data model since it is easy to use. Even if other spatio-temporal data models are available, such as the data model proposed by this study, historical data recorded as snapshots must be transferred to the new spatio-temporal data model. Chapter 3 proposes complete workflows for importing data from snapshots. This is implemented by the prototype system discussed in Chapter 4.

However, the *snapshot model* does have some shortcomings. First, snapshot is not used for change tracking, since how things change between two snapshots is not recorded explicitly. Second, if the snapshots are taken at regular intervals, the status of an object between two snapshots is unknown; hence, its value can only be interpolated. Third, redundancy impedes the model's storage efficiency because an object must be recorded when a snapshot is taken, even if nothing changed on this object.

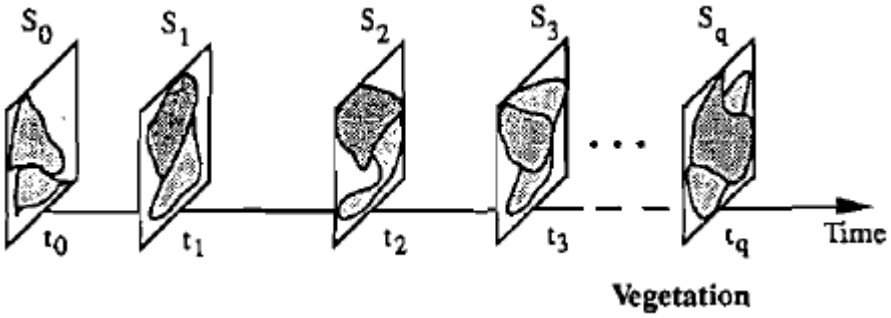


Figure 2.2 Snapshot model

(Peuquet and Duan 1995)

In Geodatabase, a scheme referred to as versioning allows users to edit the same data without applying locks or duplicating data (Zeiler 2000). A version represents a snapshot in time of the entire Geodatabase. A Geodatabase can have many versions in the same way that it is possible to have multiple snapshots to represent the study area at different times. Every ArcSDE Geodatabase has a default version— known as DEFAULT—that is the published version of the database, representing the current state of the study area being modeled.

2.2.2 The simple time-stamping model

Another simple approach is to tag every object with pairs of timestamps that indicate the time of creation and cessation of a state of this object. A special value, such as "NOW," "CURRENT," or "NULL," can be assigned to the cessation time of an object if it exists currently (Renolen 1996). This is similar to the time interval concept (Frank 1998) that records the duration for which an object maintains the same state. By applying the time interval concept, data redundancy can be largely reduced. This study proposes a *Space-Time Interval* class using an approach similar to that of the *time-stamping model*, except that a Space-Time Interval holds both the spatial and temporal information of an object.

Simple time-stamping model is class-oriented. Different states of objects at different time periods are stored in a common feature class. When people need to specify the "what" component in the spatio-temporal query, all records that belong to this "what" have to be selected first. The geometry or attributes will then be analyzed. In addition, simple time-stamping model is event-based. An event could trigger a new record of the changed object in the feature class.

The *time-stamping model* is applied in ArcGIS 10 released by ESRI in 2010 (Otto 2010). In this version, a layer can be specified as a *time-awareness layer*. A time stamp, or time interval, can be added to each feature indicating its duration of existence (Figure 2.3). This is actually an application of the *simple time-stamping model*. Users could use a time

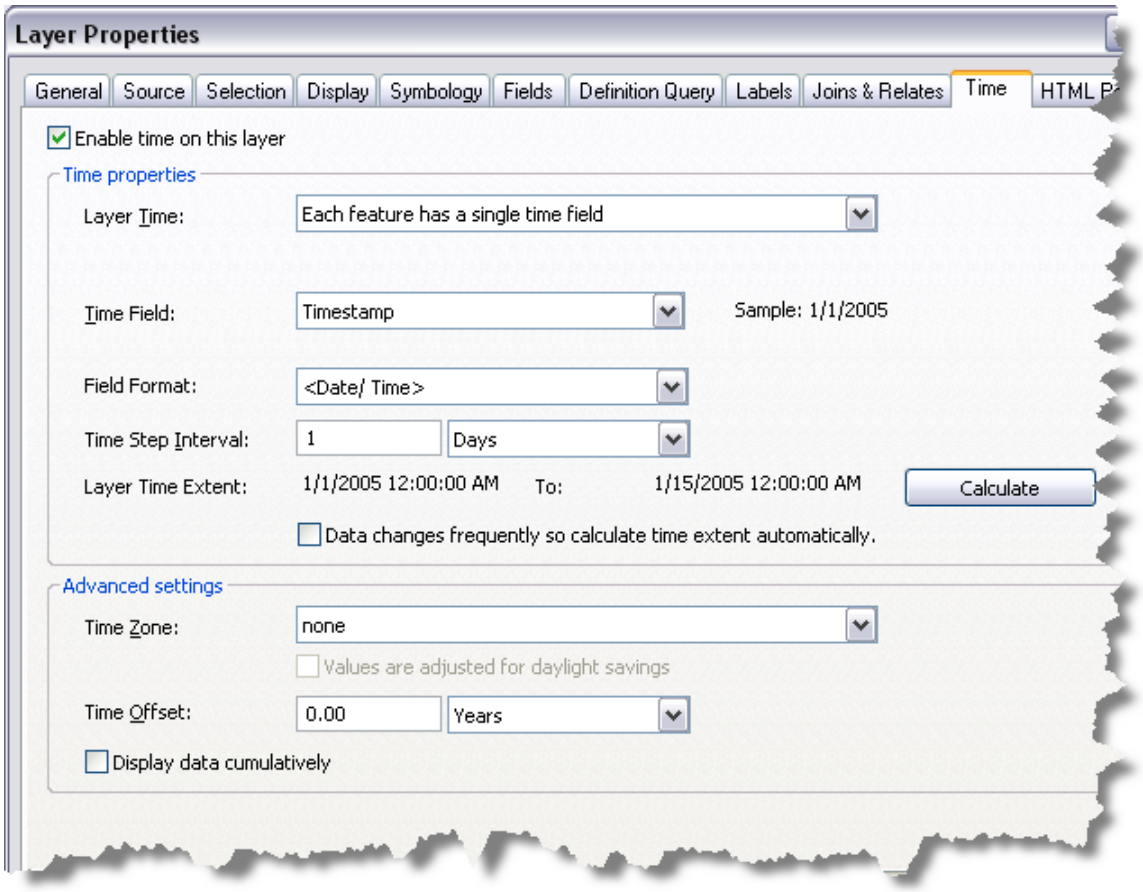


Figure 2.3 The time tab in the layer properties dialog in ArcMap 10

(Otto 2010)

slider to view all features that exist at the specified time point or time interval. However, spatio-temporal query and analysis are not available in ArcGIS 10.

One major shortcoming of the *simple time-stamping model* is that it stores different objects and different versions of the same object over several non-related tuples in the same table, which is insufficient for tracking the history of one single object since the program would have to select all records of this object at the beginning. This thesis research solved this problem by adopting the object-oriented approach; different versions of an object are stored within a *Space-Time BLOB* of this object, instead of being managed with other objects under a common attribute table.

2.2.3 The space-time composite model

Langran (1992) proposed the space-time composite model based on the principle that objects in space at different times can be projected down to a spatial plane and intersect with one another to generate the greatest common units. Each greatest common unit stores the attributes within its boundary over time (Figure 2.4). Changes can be easily

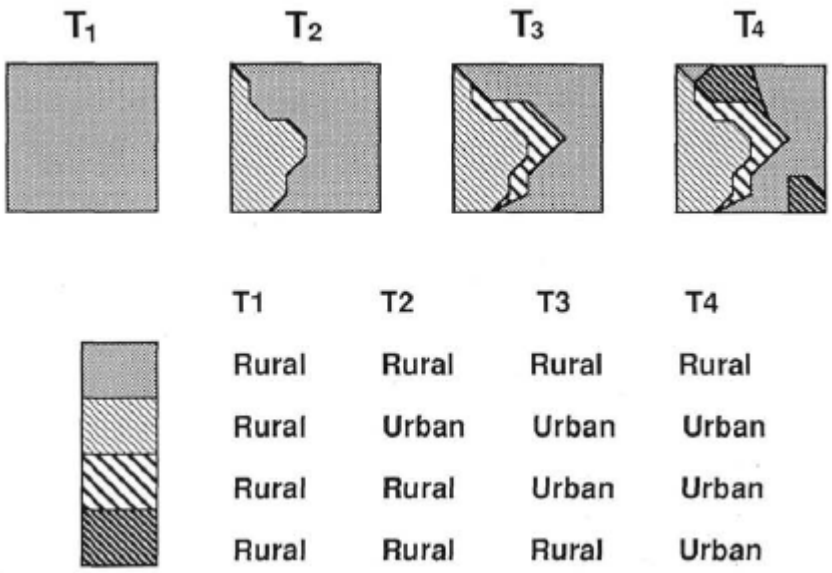


Figure 2.4 The space-time composite model

(Langran 1992)

tracked at the greatest common unit level. However, since objects in space at different times might need to be split into smaller units with new IDs, change tracking at the object-level is difficult (e.g., how land parcel A changes between 2000 and 2010).

The *space-time composite model* is could be implemented using either object-oriented approach (i.e., the spatio-temporal information of an object is aggregated and associated with the greatest common unit) or class-oriented approach (i.e., the spatio-temporal information of a greatest common unit is stored in a common attribute table with other greatest common units. Furthermore, this model could be event-based or non-event-based in terms of whether the objects are recorded by event or not.

The *space-time composite model* was adopted in this study to handle objects without entity identity. Identity is a unique character that distinguishes one object from another; land use data is an example of objects without entity identity. This kind of data does not keep a consistent ID through time for each object so it cannot be recorded at object-level. However, managing spatio-temporal information and tracking changes at the greatest common unit level is simple and straightforward.

Nevertheless, the *space-time composite model* is cumbersome when dealing with data that is continuously updating. The reason is that the previous greatest common units layer has to be projected and intersected with the latest layer to generate a new greatest common units layer. When handling very large real-world data that update at a high frequency, the *space-time composite model* is inefficient. However, addressing this issue is beyond the scope of this thesis research.

2.2.4 The amendment model

The key concept in *amendment model* (Langran 1992) involves the "amendment vectors" that explicitly record spatial changes over time. Any increment is stored based on the status of previous time point (Figure 2.5). The temporal information of this change is stored as attributes of the increment vector. This approach satisfies a variety of spatio-temporal query functions by retrieving and manipulating recorded amendment vectors, such as how a polygon changes at a given time point.

Since the *amendment model* explicitly records the geometry change of an object over time, the object-oriented approach could benefit its performance by recording the changed geometry in sequence for each object. Also, it could be event-based or non-event-based in terms of whether each object is recorded by event or not.

The *amendment model* is not suitable for objects without entity identity since the state of each object at each time point must be known in order to calculate the amendment vectors. Furthermore, the amendment model cannot store attribute change that occurs without spatial change since attribute change does not generate amendment vectors. In this thesis research, the proposed space-time GIS model gives attention to this issue so that changes in space, attributes, or both space and attributes can be captured.

2.2.5 The grid model

Langran (1992) proposed a *grid model* for raster type data. Every cell has a list of historical values in this approach. A new value is added to this list only if change occurs (Figure 2.6). The latest value is always on top, so the current status of the study area can easily be accessed.

The *grid model* applies the object-oriented approach. The changed value of a pixel is stored with the pixel. As a result, tracking the change of the pixel is efficient. Whether it is event-based or non-event-based depends on how the images are captured (i.e., by event or regularly).

This approach improves on the *snapshot model* in that unchanged pixels are not recorded so that the data redundancy is greatly reduced. However, the time point at which change occurs for each pixel cannot be determined precisely if data are recorded at regular intervals, unless event-based approach is taken such that data can be recorded when change occurs. Therefore, the *grid model* is only a somewhat an improved version of the snapshot model. This thesis research proposes a method for spatio-temporal raster data storage in which each pixel is associated with a Space-Time BLOB that precisely records the attribute changes of this pixel over time.



Figure 2.5 The amendment model

(Peuquet and Duan 1995)

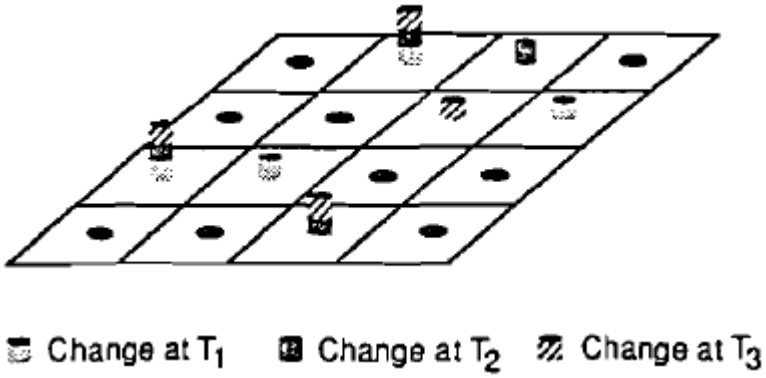


Figure 2.6 The grid model

(Peuquet and Duan 1995)

2.2.6 Event-based spatio-temporal data model (ESTDM)

Peuquet (1994) suggested a *TRIAD database framework* to integrate time into GIS with an implementation designated as the *event-based spatio-temporal data model* (ESTDM) (Peuquet and Duan 1995). ESTDM uses concepts from the snapshot (Armstrong 1988) and grid model (Langran 1992). Dual run-length-encoding is applied in ESTDM to maximize storage efficiency. To cover both temporal and spatial domains, ESTDM only records the study area when a specific event happens (e.g., land use change), and keeps track of each new cell value and the pixel location where a specific new value is assigned (Figure 2.7). Therefore, the total amount of storage is determined by the number of events and the number of new values at each event. The event-based concept makes change tracking with ESTDM highly efficient. ESTDM applies the class-oriented approach. Instead of storing new values at the pixel level, the whole area with the same new value is recorded together.

Nevertheless, ESTDM is not applicable to vector-based data; in addition, only one variable can be represented because a raster layer is associated with only one theme. The space-time GIS data model suggested by this thesis research has the potential to

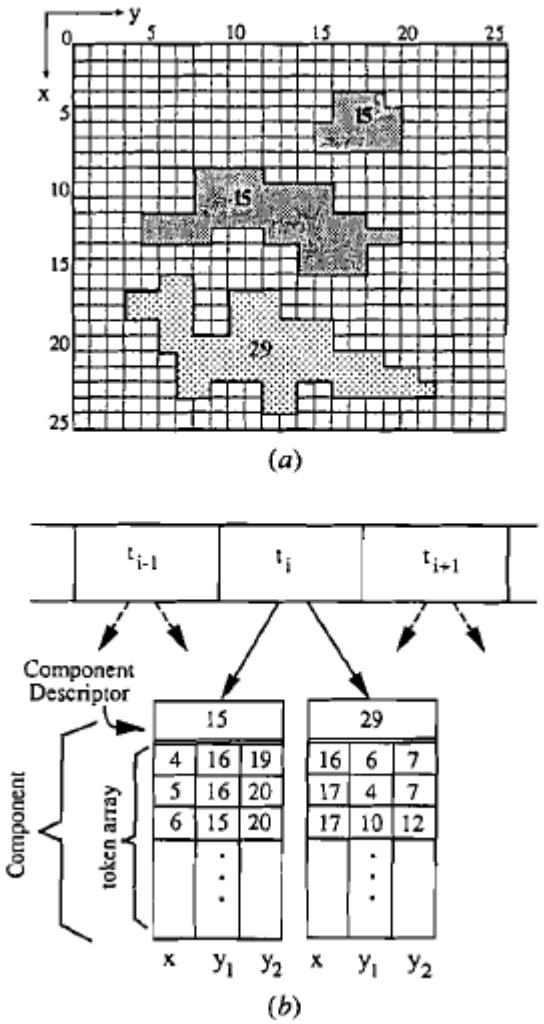


Figure 2.7 Event-based spatio-temporal data model

(Peuquet and Duan 1995)

overcome this shortcoming by assigning a separate array for each theme at each pixel. An array records how the value of a theme at this pixel changes over time.

2.2.7 Three-domain model

Yuan's *three-domain* (Yuan 1999) representation covers the semantic, temporal, and spatial domains, along with domain links. The semantic domain represents entity classes with unique identifiers throughout the transition, the temporal domain records the time point or time interval instance, and the spatial domain represents spatial configurations of the study area over time. Domain links connect these three domains to convey meaningful information. *Three-domain model* uses class-oriented approach since spatio-temporal variation of object is kept using common domain tables and domain links, rather

than being stored separately with object itself. Also, it is event-based. The exact time when a specific change happens is recorded in the temporal domain.

Yuan (1999) used a hypothetical scenario to illustrate how the *three-domain model* works. However, it would be very complex to maintain the links between different domains for a large amount of data.

2.2.8 Object-oriented model

Worboys (1992) proposed an object-oriented spatio-temporal data model. It consists of two-dimensional spatial objects and a third temporal dimension associated with each object. The basic element in this model is the *spatio-temporal atom*. This *object-oriented model* is also event-based. New *spatio-temporal atoms* can be stacked on top of existing ones when events occur (Figure 2.8). However, non-spatial changes (i.e., attributes changes) cannot be recorded unless they are associated with changes in both space and time. Worboys (1995) later incorporated both *event time* (i.e., real-world time when something happens) and *database time* (i.e., time when something is entered into the database) to form a spatio-bitemporal model that records the existence of an object in the real world and database, respectively. He further investigated the difference and relationships of "*change*", "*event*" and "*process*" associated with the object-oriented modeling constructs (Worboys 1998). According to Worboys, an *event* is an occurrence or an outcome that has significance, while a *process* involves a series of activities, events, and interactions so that a goal can be achieved.

The data model proposed by this thesis research is also object-oriented. The identity information of the object that remains consistent over time is used to track object changes. Moreover, it supports recording attribute change without spatial change, since the attribute change can trigger an event as well.

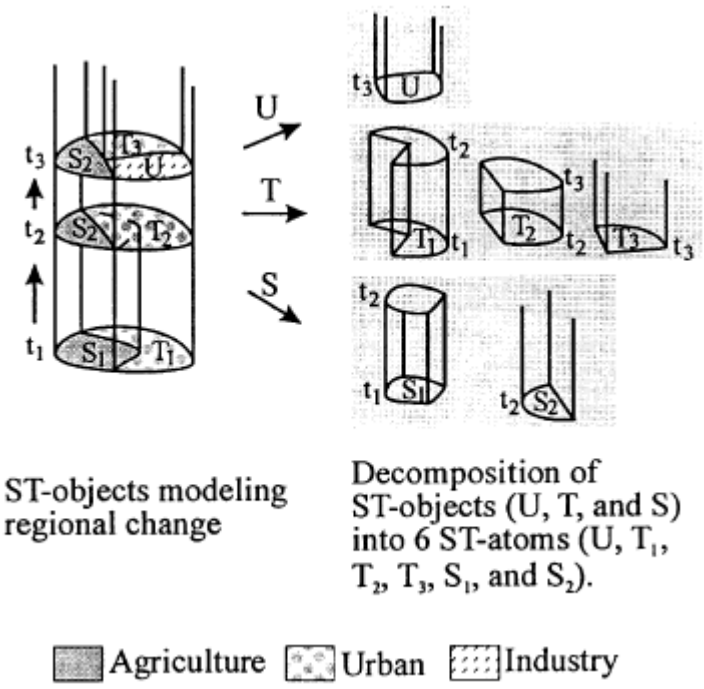


Figure 2.8 Object-oriented model

(Yuan 1999)

2.2.9 The extended dynamic GIS model (EDGIS)

Some researchers view the world as being composed of atoms, and space-time information can be derived using geo-atoms (Goodchild, Yuan et al. 2007; Pultar, Cova et al. 2010). An *extended dynamic GIS* (EDGIS) *model* was proposed predicated on this concept. Using EDGIS, the real world is abstracted as themes, features, and *space-time points* (*STP*, based on the concept of geo-atoms). An example of how real-world objects are organized using the EDGIS model designates Unita Lakes as a theme is recorded in the Theme table, which is linked with the Feature table (Figure 2.9). Each feature table contains all the features (e.g., Mirror Lake, Red Castle Lake) included in a specific theme. A feature consists of many *space-time points*, which are recorded in the Space Time Point table, along with the spatial location, temporal and attributes information of this *space-time point*. In this example, the links between these tables shows how theme, feature and *STP* are connected. Take Mirror Lake for example, it is a feature of the Unita Lakes theme. Mirror Lake has three *STP*s, each of which is a record in the Space Time Point table. Since all *STP*s are managed in the Space Time table, EDGIS uses the class-oriented approach. Moreover, it could be event-based or non-event-based depending on whether *STP*s are recorded by event or not.

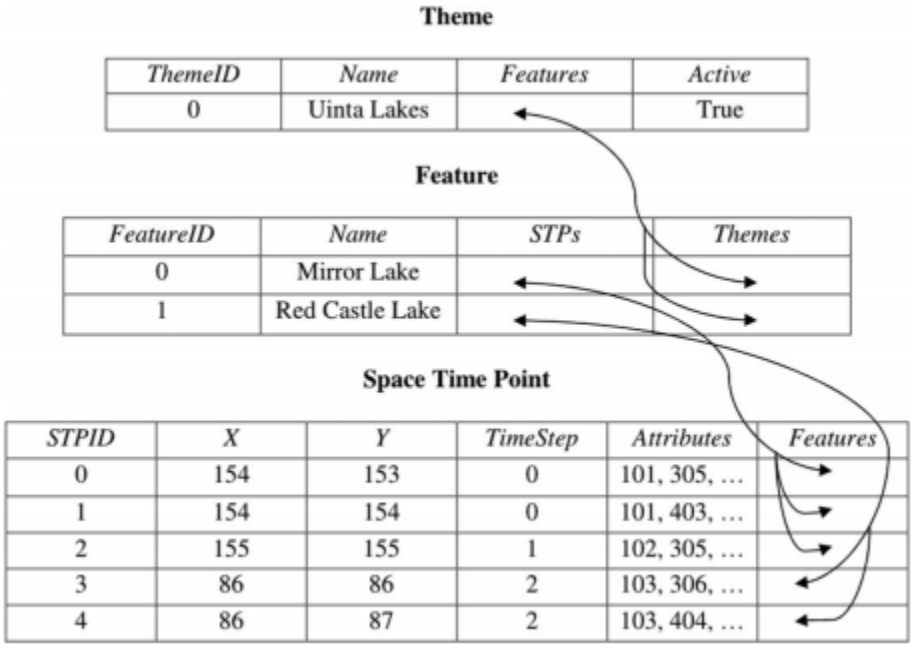


Figure 2.9 The data structure of EDGIS

(Pultar, Cova et al. 2010)

The major contribution of *EDGIS* is that it abstracts the objects in the real-world from both discrete and continuous perspectives, since *STP*s are more fundamental than objects or fields and represent a key element of either. However, such a revolutionary design requires new data collection method that is able to simultaneously capture discrete and continuous information. In addition, all *STP*s are stored in a common Space Time Point table. It is sufficient for the small datasets tested in the prototype system. However, large dataset could lead to very long Space Time Point table. This is inefficient when performing spatio-temporal queries.

2.2.10 ESRI parcel model

ESRI parcel model is designed for archive parcel transactions. It can also be used for other similar scenarios. Parcel maps in database are continuously updated over time. Real estate transactions, parcel merges and splits occur frequently. This model is able to accurately track the lineage or history of a parcel as it changes by storing changes in additional tables, which allow the historical representation of the updated features. Figure 2.10 illustrate how this parcel model works. The TaxParcel feature class represents the latest state of land use parcels, while the TaxParcelHistory feature class records the historical land use parcels. In_Date is the time when a land use parcel was created, while Out_Date is the time when it was removed. In January 30, 1994, a land use parcel with ID 200 existed. In July 16, 1997, parcel 200 split into parcels 201 and 202. Parcel 200 is recorded in the TaxParcelHistory feature class with an Out_Date

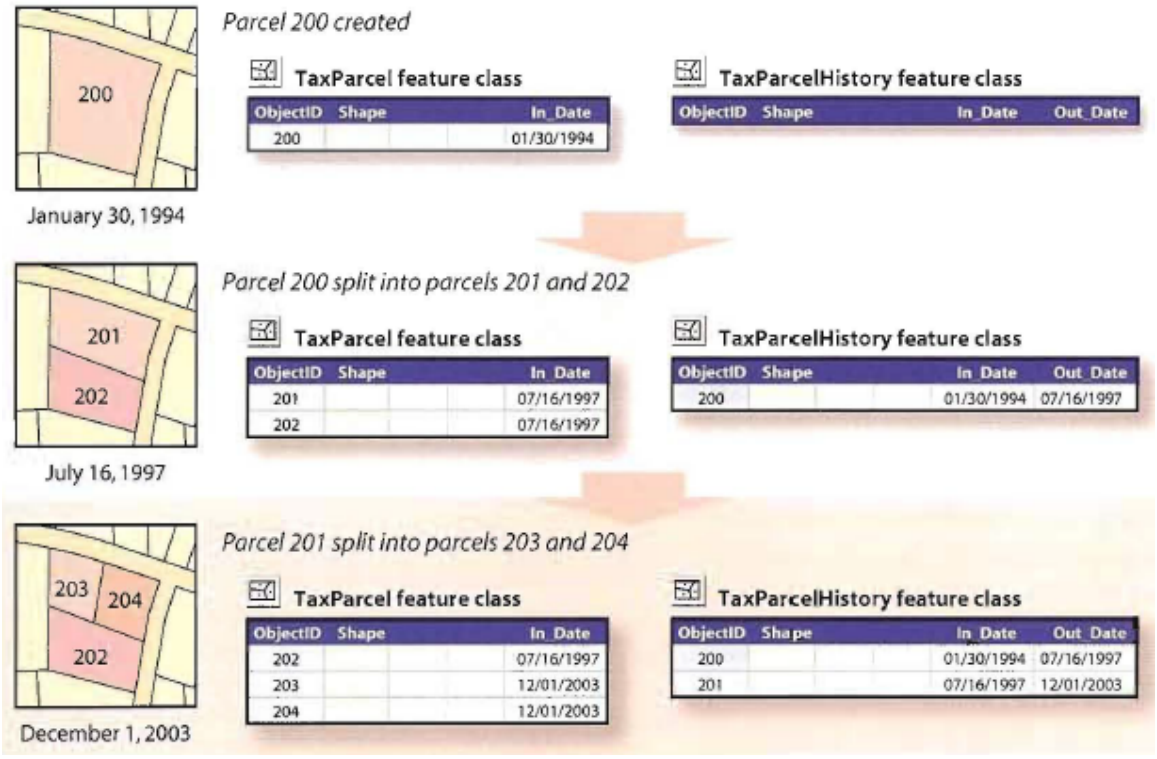


Figure 2.10 ESRI parcel model

(Arctur and Zeiler 2004)

07/16/1997. Land use parcel 201 further split into parcels 203 and 204 in December 1, 2003 (Arctur and Zeiler 2004).

ESRI parcel model is class-oriented. All current objects are stored in a feature class which represents the latest state of the study area, while all objects that have been removed are archived in a historical feature class. This model applies the event-based approach. Changes in the database can be made once the real-world transaction occurs.

One major shortcoming of *ESRI parcel model* is that people might need to spend a lot of time in switching between current table and historical table in order to track changes of a certain object. For example, to understand the history of parcel 204, we need to know the In_Date of parcel 204 (12/01/2003) in the current feature class, and then access all records in historical feature class with Out_Date 12/01/2003. That is parcel 201. After that, we have to access the "current" feature class with the same In_Date as parcel 201(07/16/1997) and use the same procedure to know the history of parcel 201. This process is cumbersome, especially when an object has a long change history.

2.2.11 The TimeObject in the national cooperative highway research program (NCHRP) and other types of time

The previous 10 spatio-temporal data models handle both spatial and temporal domains. The *TimeObject* and other related topics in this section consider only temporal domain, with a great emphasis on different forms of time.

NCHRP is a program involving problem areas of highway planning, design, construction, operation, and maintenance in the US (Adams, Koncz et al. 2001). In NCHRP 20-27(3), recording and making use of temporal information with spatial object is one of the key topics. The storage of the temporal element of phenomena and its behavior is represented as a time object (Figure 2.11). In NCHRP 20-27(3), time object is the temporal equivalent of a spatial object, where the time object provides the quantitative description of temporal characteristics of transportation features and events.

The *time object* in *NCHRP 20-27(3)* is far more complex than a time stamp (time point). Other forms of time are considered and incorporated, such as time interval consisting of a start time, an end time, and the length between them. *Time aggregate* is a combination of time objects to form a temporal group. *Time aggregate* is inherited by four subclasses: *time cycle*, *time break*, *time stage*, and *time sequence*. A *time cycle* has its end point connected with its beginning point to represent repeating behaviors, such as a weekly class schedule. A *time break* consists of the period of time represented by the difference in two disjointed, non-overlapping intervals. A *time stage* is made of two or more intervals that can meet, overlap, equal, start, end, or be disjoint. A *time sequence* consists of two of more time intervals that meet.

This study is inspired by *NCHRP 20-27(3)* in terms of different forms of time and their relationships. Although temporal structures, such as *time cycle*, have not been implemented, they will be incorporated within the proposed space-time GIS data model in the future. Instead of considering time object as separated from spatial object, the data model in this thesis research stores both spatial and temporal information together as Space-Time Elements within the Space-Time BLOB.

Similar temporal forms were proposed by Frank (Frank 1998), who also suggested the *time branch* (Figure 2.12). *Time branches* are useful in representing more than one temporal arrangement. For example, a student may have three different plans after 6 pm: studying, exercising, and playing. Consequently, his timeline bifurcates into three time branches after 6 pm. Similarly, *time branches* can also be used to depict possible historical periods. Incorporating *time branches* in the proposed data model is discussed in Chapter 5.

As basic temporal components, time points and time intervals can be definite or indefinite. According to Young and Ziman (Young and Ziman 1971), an indefinite time point or time interval does not have a quantitative measure. We only know its previous and next time points or time interval. This is not uncommon in daily life; for example, one knows that he or she encountered a friend on the way to class but cannot remember

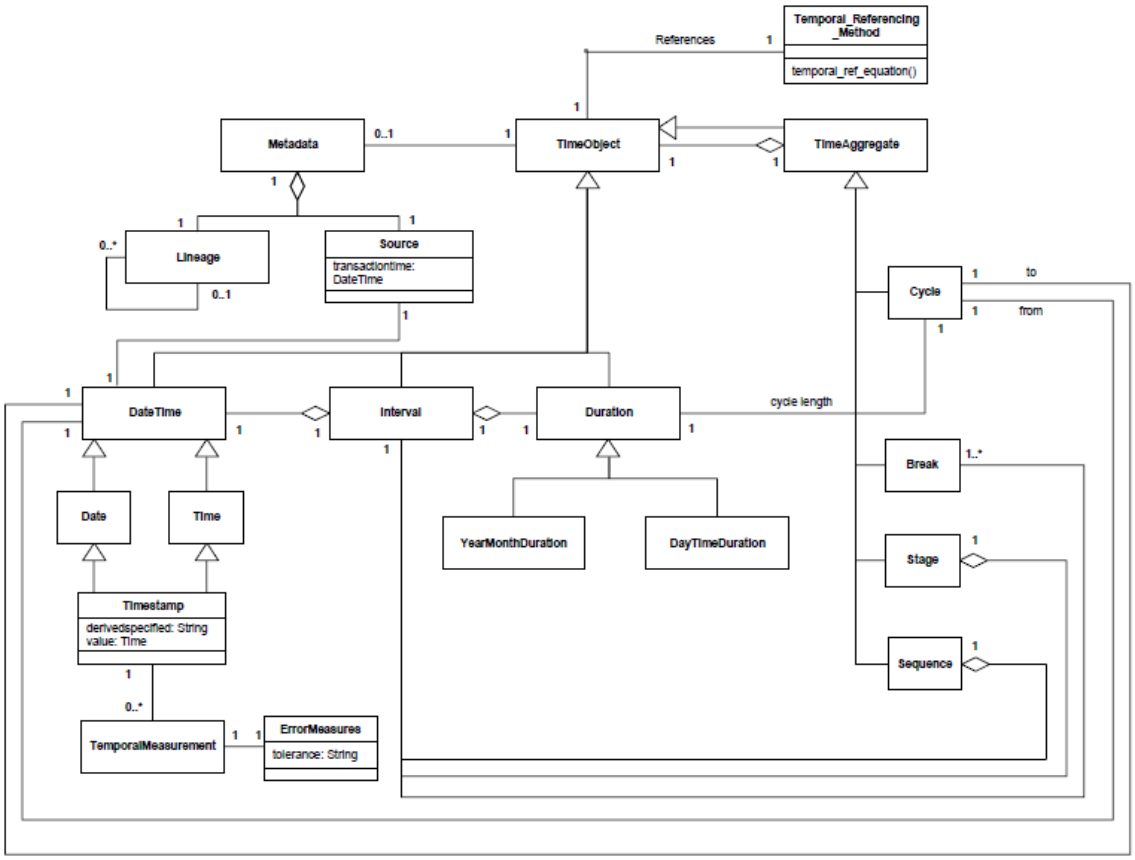


Figure 2.11 Time object in NCHRP 20-27(3)

(Adams, Koncz et al. 2001)

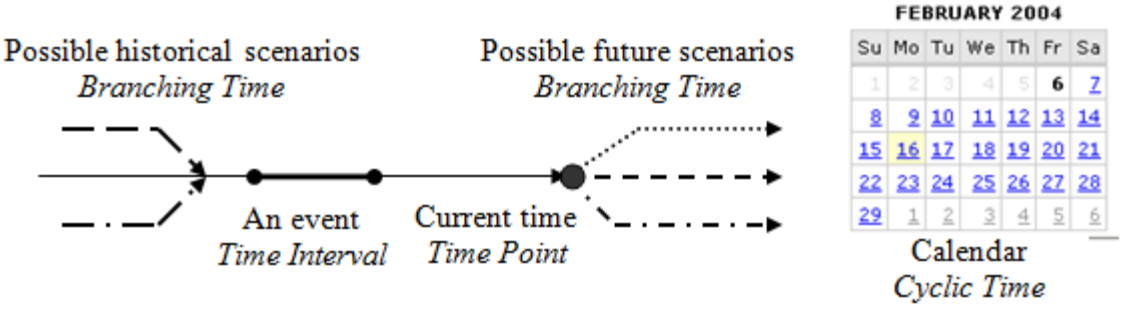


Figure 2.12 Different types and structures of time

(Frank 1998)

exactly what time it was and how long their conversation lasted. In this case, the encounter may still be incorporated in the time line using the concept of indefinite time

point or time interval. Consideration of indefinite time types is not a feature of this thesis research, but they will be incorporated in a future study due to its importance.

2.3 Spatio-temporal Visualization

Spatio-temporal data, which involves the variation in both spatial and temporal dimension, are usually far more dynamic and complex than static data. People might be easily confused by the overloaded spatio-temporal data using the traditional visualization approach. On the other hand, a dynamic, interactive visualization approach could benefit research and facilitate understanding of the overall processes, patterns, and trends of spatio-temporal data, as well as the result of the spatio-temporal query.

The Exploratory *Spatio-Temporal Analysis Toolkit* (*ESTAT*) developed by the GeoVISTA Center at the Pennsylvania State University (Robinson 2005) supports the visualization of spatio-temporal data using several representation methods (Figure 2.13). A *scatterplot* can be used to compare the values of two variables, with one plotted against the other, in order to disclose the correlation. By assigning each location in the *scatterplot* a color, *ESTAT* could draw a *bivariate map* to represent the characteristics of the study region. A *parallel coordinate plot* (*PCP*) is designed to display multi-dimensional data by converting data categories into axes and then drawing lines from one category to another based on the values at each category. A *time series graph* illustrates the variation of a single variable over time. The most attractive feature of *ESTAT* is that these four elements are not isolated; instead, they are dynamically linked. If user brushes over a point on the *scatterplot*, a feature on the map, or a line on the PCP or the time series graph, the corresponding subjects in all other elements are automatically highlighted.

Another challenging topic in geo-visualization is how to represent spatio-temporal multivariate data in a way that people can understand. Guo et al. (2006) applied the self-organizing map (SOM) to transform multivariate data into a two-dimensional space. A two-dimensional color scheme was utilized in conjunction with the SOM to provide meaningful information, upon which it is possible to discern the temporal pattern of multiple variables.

This thesis research focuses on the spatio-temporal data model and spatio-temporal query. However, an interactive representation of the spatio-temporal phenomenon is extremely helpful to suggest ideas about the patterns, trends, and processes of the themes and variables under consideration. This could provide a topic for future research.

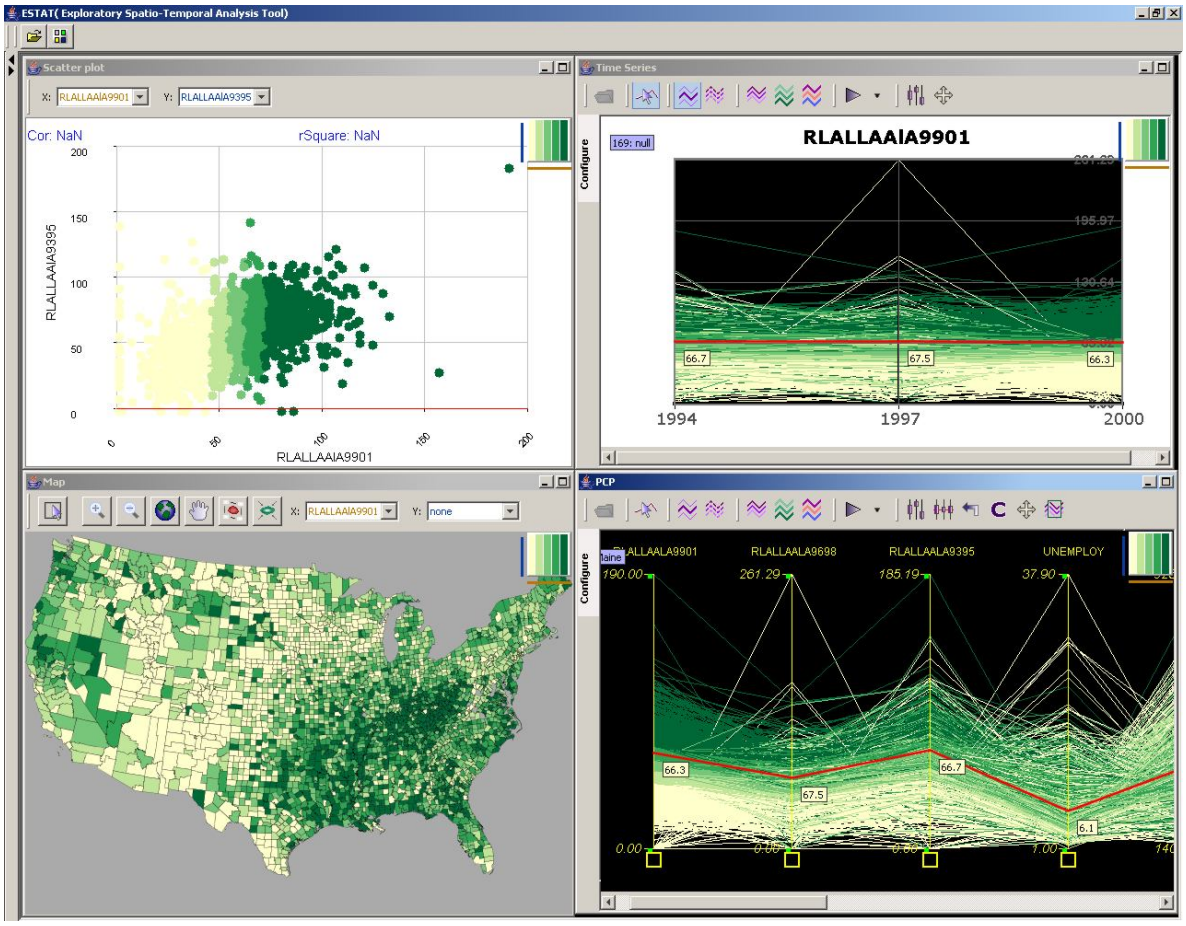


Figure 2.13. Four representation methods in ESTAT

(Robinson 2005)

2.4 Moving Objects Database

Research on moving objects has been popular in computer science since the late 1990s, especially when location-aware devices began producing a wealth of position data (Schneider 2009). Challenging moving objects database research topics involve updating continuously changing data in a database efficiently, designing a capable query language (Wolfson, Xu et al. 1998), and discovering hidden patterns in the behavior of multiple objects in large data sets (Laube, Imfeld et al. 2005), among others.

In some cases, only time-dependent locations are of interest, in reference to *moving points*. Typical examples of this include persons, vehicles, aircrafts, and so on. Such types of moving objects change positions continuously in space over time; hence, Hägerstrand's *time geography* concepts (Hägerstrand 1970) could be useful, especially when there is a need to represent the constraints of individuals (Shaw and Yu 2009). The classical *space-time prism* can be refined or coarsened in terms of the granularities forming *threads*, *trace*, and *necklaces* (Hornsby and Egenhofer 2002). A *thread* refers to

an ordered sequence of space-time samples that resembles *space-time path* in *time geography*. A *trace* is a coarsened thread with only one start point and one end point. A *necklace* is a version of space-time prism refined by adding intermediate sample points inside the prism.

In other cases, time-dependent shapes and/or areal extents that grow or shrink through time, namely moving regions, must be handled, examples of which include hurricanes, forest fires, and so forth. In rare cases, time-dependent shapes and/or linear extents (moving lines) that can lengthen or shorten, exhibited by snakes and streams, is of interest. The spatial dynamics of linear and polygon objects are more difficult to represent in the traditional non-spatial relational database since such objects could not be recorded by a single coordinate pair. Erwig et al. (1999) proposed a new line of research to incorporate moving regions as 3D (2D space and 1D time) entities. These types of entities are abstracted as data types supported by a variety of database management systems (DBMS).

On the other side, moving objects research could be classified as historical moving objects research or predictive moving objects research. The former aims to describe the continuous, dynamic, and time-dependent behavior and location change of spatial objects over time. Thus, spatio-temporal analysis could be performed. The latter intends to predict the temporal evolution of spatial objects at the present time and in the future. The *moving objects spatio-temporal model* (*MOST*) is the only model so far to describe future location of the moving objects (Prasad Sistla, Wolfson et al. 1997; Prasad Sistla, Wolfson et al. 1998). Using this model, the predicted position of the moving objects is represented by motion vectors. An update is only needed when the object's deviation is beyond the specified threshold. A query language, *future temporal logic* (*FTL*), which includes both spatial operators (e.g., object INSIDE polygon) and temporal operators (e.g., UNTIL, EVENTUALLY in the future), makes it possible to conduct spatio-temporal queries on this model (Wolfson, Xu et al. 1998).

Both moving objects databases and GIS-based spatio-temporal data models are intended to store the spatial and attribute information of the real-world objects over time. The most significant difference is that the former uses commercial DBMS, while the latter is based on GIS data structure and database, such as Geodatabase. In addition, the GIS-based spatio-temporal data model can take advantage of various spatial analysis functions in GIS platform. Queries that require geometric operation, such as "what is the geometric difference of land parcel A between 2000 and 2010" could be easily conducted with GIS-based spatio-temporal data models. In this thesis research, the proposed data model is GIS-based. As a result, time-dependent locations and shapes are all addressed since points, lines, and polygons are all recorded as Geometry type in GIS packages. This thesis focuses on historical objects. In the future, predictive research of the object could be conducted based on the understanding of the trend and process derived from the historical data.

In conclusion, this chapter explores topics relevant to this study that have been amassed from previous research. Different types of spatio-temporal query were reviewed first, followed by a detailed discussion of spatio-temporal data models which are used to support spatio-temporal query. Spatio-temporal visualization methods which benefit the understanding of the spatio-temporal data and spatio-temporal result were also reviewed in this chapter. Moving object databases study that shares some similarities of spatio-temporal data models are investigated in the final part of this chapter.

CHAPTER 3 AN OBJECT-ORIENTED SPACE-TIME GIS DATA MODEL

Many efforts have been made to achieve a powerful space-time GIS data model in previous decades, such as the snapshot model, event-based spatio-temporal data model to support various kinds of spatio-temporal queries. Current space-time GIS models all exhibit some strengths and weaknesses. Some models can handle only vector or raster data, some models have serious data redundancy, while others might be too difficult to implement so as to solve real-world problems. In this chapter, a conceptual model is proposed for processing spatio-temporal datasets and to organize features that evolve over time in the Geodatabase. The specific design in unified modeling language (UML) and the processing flowchart are presented in this chapter. This study focuses on polygon type data; however, this space-time GIS data model is capable of processing data types other than polygons, such as points, polylines, and rasters.

3.1 Relevant Conceptualization of Space and Time

3.1.1 The prevalence of the snapshot model and its limitation

Time-stamping is a traditional way to introduce the temporal dimension into GIS. A snapshot records the state of a certain area under study at a specific time point. However, the snapshot model can result in serious data redundancy, especially when the number of snapshot is large. In order to store data covering 50 years in the land use scenario, the analyst might need to deal with about 2,600 layers. It is not uncommon that many parcels do not change for a very long period, while they are still recorded as standalone features in each snapshot layer. Although storing hundreds of thousands of layers is not a problem for Geodatabase, such data structure in itself is inefficient, which may lead to poor performance in spatio-temporal query and analysis. There is no reason to store a feature each time a snapshot is taken of the entire study area when this feature remains exactly the same between two snapshots. In fact, it is more efficient to store a feature only once until a change of either geometry or attribute occurs.

Another reason that the snapshot model is not appropriate for space-time GIS is that it is impossible to keep track of spatio-temporal changes when the snapshot model simply records an object without maintaining its lineage. Even if we have a snapshot at each point in time when a change occurs, the lineage of information is still lost since the snapshot model does not explicitly store which object changes at which point in time.

The essential reason of the incapability of snapshot model is that it records data according to a specific division scheme of the timeline, either periodically or in response to an event, rather than focusing on the object itself. This current study refers to that method as a timeline-oriented approach due to the fact that it emphasizes the lapse of time but ignores how things evolve over time. Each object is not considered to be a continuation

of its ancestor's existence. Consequently, recorded data cannot track how objects change over time.

3.1.2 Object-oriented view of space and time

This study adopts an object-oriented event-based approach that concentrates on variation in objects rather than on the lapse of time as the timeline-oriented snapshot model does. Each object in the real world would be recorded when a significant change applies on it, rather than being recorded again and again with the advancement of the time.

A more efficient way to store continuous and consistent spatio-temporal information is to assign a start time and an end time for the feature during which its geometry and attributes do not change given the specific spatial and temporal resolutions. The resolution concept is very important: If the resolution becomes finer, there is no guarantee nothing would happen. For example, if we observe a student each hour, we may find out that he was in the classroom at 8:00 am, 9:00 am, and 10:00 am. When the temporal resolution is set to 5 minutes, the result might not be the same. He could go to the room, leave to grab some snacks from the shop, and return during that time. This is similar to the modifiable areal unit problem (MAUP) in spatial analysis research (Dark and Bram 2007). Therefore, when using the approach proposed in this study, it is important to pay particular attention to the temporal resolution.

A start time and an end time, which together constitute a time interval, are the key attributes in tracking spatio-temporal changes. A start time specifies the beginning time point of a new state of the feature, while an end time indicates the time point when it acquires a new status. No record would be created for a feature again during a start time and an end time. However, this approach is not that efficient when people need to retrieve a snapshot at a given time point. The state of each object at the specified time point needs to be generated by figuring out which time interval this time point falls on.

It is not easy to access the status of each feature in the snapshot layer method because the intermediate status of a feature is probably unknown. However, retrieving states from time intervals is relatively simple as long as the time interval that the specified time falls in can be found. This is identical to the "during" operation in the temporal topology proposed by Allen (1984).

However, it is not possible to use only time intervals to represent the existence of the object. In many cases we may know an object exists at one time point without having any idea about whether it does thereafter. Thus, both time point and time interval data are necessary. For example, if a pop star was spotted at an airport at 9:00 pm and he took part in an activity from 10:00 pm to 11:00 pm. To record his presence into a database, both time point and time interval data structures are useful.

Since object-oriented space-time GIS design focuses on each object, it can record spatio-temporal changes precisely. Different objects might have different configurations of time

points or time intervals. Therefore, the number of time elements and the time span of time intervals of each object might not be the same, since different objects change at different times. For example, Object A could have time intervals of $I_{A1}[T_1, T_2]$ and $I_{A2}[T_2, T_3]$; Object B could have a single time interval of $I_B [T_4, T_5]$. The number of intervals, as well as the duration of intervals for Object A and Object B are different in this case.

A critical problem is that sometimes historical data are organized as snapshot layers. As noted above, a snapshot is taken in response to events or at regular intervals. On the one hand, if the snapshot is event-based, although the exact change time is recorded we still have no idea about which object changes at that time since snapshot model does not record the ID of the changed objects. On the other hand, if the snapshot is taken regularly, such as once a week in the land use scenario, we are unable to determine the exact time of change. In both situations we lose track of historical changes for all objects. In other words, the way in which most historical data are organized does not allow change tracking. We can still determine the time interval for each consistent status of each feature by assigning the start time and end time extracted from the snapshots. However, it should be noted that the start and end times are probably not the moments when changes take place on a specific object.

3.2 Design of the Object-oriented Space-Time Class

The previous section briefly discusses the characteristics of time and how to conceptualize it so as to represent spatio-temporal information in Geodatabase. This proposes an object-oriented Space-Time GIS design.

The term *object-oriented* is used here to describe this space-time GIS design from two perspectives. First, as discussed above, change in object over time is more important than change in time with objects. The object, referred to as *feature* in Geodatabase, is the core of this design. Second, space and time are abstracted into classes, and the object-oriented principle is used to implement this design (Figure 3.1).

The Snapshot Feature Class represents the feature class that records the historical states of each object within the study region at a given time point. Thus, every Snapshot Feature Class is associated with a DateTime which indicates when this snapshot is taken. As explained above, since the snapshot model does not allow change tracking, the Space-Time Binary Large Object (BLOB) recorded using the proposed model could not precisely reflect when an object changes over time. However, after the spatio-temporal data are organized into Space-Time BLOB using this proposed model, change that occurs in the future could be recorded precisely.

There are two subtypes of Snapshot Feature Class: Snapshot with Entity Identity Class and Snapshot without Entity Identity Class. The term *identity* is the property of an object that distinguishes it from all another objects, even if they have the same attributes. A typical example of object with entity identity is the map of the United States. In this

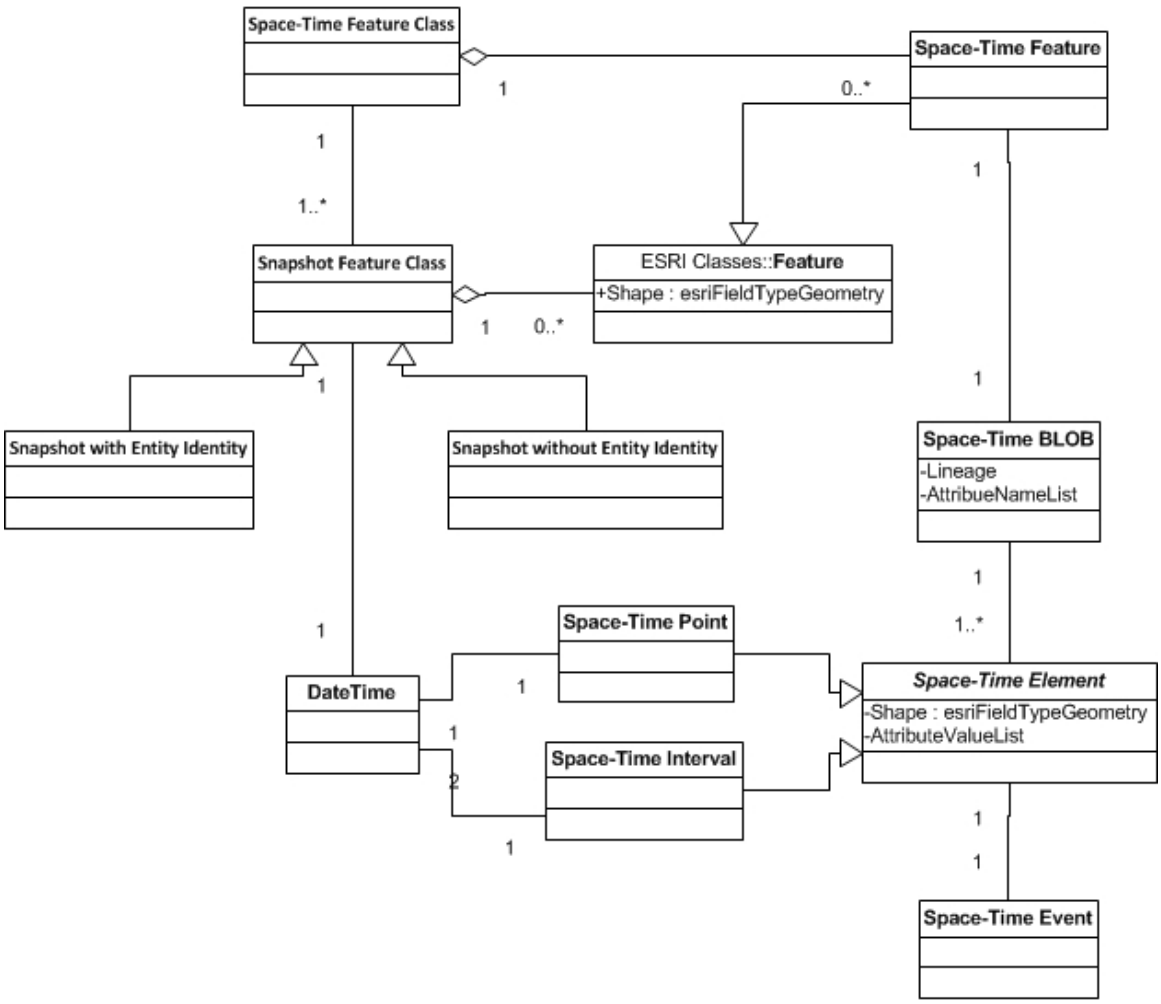


Figure 3.1 The UML diagram of the proposed space-time GIS design

map, Tennessee is an object with entity identity; the geometry of the entire state has political meaning. Locations and objects that are within the state boundary share certain common attributes. People in Tennessee are subject to the same taxes and should obey the same state laws. A person is also an object with entity identity since everyone possesses a unique existence in the world. Even if twin brothers look the same, we still say each of them is unique. On the other hand, one cannot tell the difference between one object without entity identity from another when they have same attributes. For example, in the land use map we cannot distinguish one forest parcel from the other, except for their positional differences, because they are without unique identity.

One problem when relying on identity property is that identity can change over time. For instance, the Chinese capital of Beijing was once called Bei Ping. In this study, we consider the object with a new identity to be different from its ancestor. Moreover, an object could be split into separate parts, in which case, the separated part which holds the old identity is treated as a continuation of its ancestor. Others are considered as new

objects. In addition, an object could be merged with other object(s) to form a new one. In this case, if the new object has the same identity as one of the objects it is merged from, the new object is treated as a continuation of the old object with the same identity. However, if the merged object has a new identity it should be regarded as a new object.

In the Snapshot with Entity Identity Class, each object is understood to have its own distinct existence. However, in the Snapshot without Entity Identity Class, each object only represents a location of an area in the real world without any unique identities. It should be noted that it is necessary to distinguish between the Snapshot with Entity Identity and the Snapshot without Entity Identity classes since the internal mechanisms used to process and create the Space-Time BLOB from these two different types of snapshots are different. Moreover, each Snapshot Feature Class can contain from zero to many objects, while an object belongs to only one Snapshot Feature Class.

At the top of this UML is a Space-Time Feature Class. This is the key component in the Space-Time GIS design as the Space-Time BLOB is stored with this feature class. The Space-Time feature class deals not only with objects in one snapshot feature class, but all objects that once existed in the Snapshot Feature Class would be recorded in this feature class as its initial status, no matter how long in history they exist, or how many times of changes they undergo. The Space-Time Feature Class treats the object that exists only in one snapshot in the same way as the object that exists in all snapshots. The object with a longer period of existence does not possess any precedence or special characteristics in the Space-Time Feature Class. The Space-Time Feature Class contains STBLOB, a field used to store the Space-Time BLOB for each object. Also, each Space-Time Feature Class contains from zero to many Space-Time Features, which are special types of features. In addition, one Space-Time Feature stores one Space-Time BLOB.

A Space-Time BLOB contains the spatio-temporal information of the object it belongs to. It is referred to here as a Space-Time BLOB because a Space-Time BLOB instance needs to be serialized into a binary file and written into a BLOB type field in the Space-Time Feature Class, STBLOB. BLOB is a special data type widely used in most commercial database software. It is usually used for storing media files, such as images, audios, or videos. In fact, BLOB can store any file type as long as it can be converted into binary format. Geodatabase also supports BLOB (Zeiler 2000). This provides a perfect method to store serialized binary space-time objects. The Geometry field records the geometry of an object at a time, while the STBLOB field stores the spatio-temporal information of an object over time. A Space Time BLOB stores the names of all the attributes an object might have. For instance, the State of Tennessee might have attributes like population, GDP, and so on, values that might be different in each snapshot. The Space-Time BLOB for Tennessee State has a list of attribute names (i.e., population, GDP, etc.). However, it does not have any specific value which is handled by Space-Time Element. A Space-Time BLOB also stores the lineage information for objects without entity identity. For instance, by checking the lineage information, a user might find out that a wasteland parcel in 2000 was a forest parcel in 1990.

A Space-Time BLOB serves as a container of one to many Space-Time Element(s). A Space-Time Element can be formalized as a tuple:

$$<t, g(t), a(t)>$$

Where *t* is the time of the Space-Time Element, the time here can be a time point or a time interval during which the geometry and attributes of the feature are constant. In some cases, the Snapshot Feature Classes might hold many attributes that are not significant. It might be necessary to specify several key attributes that could trigger an event when one of them changes value. As a result, an object is considered to be unchanged when a value of its insignificant attributes changes. In the formula $g(t)$ represents the geometry of the object at time t. It might be totally different than the geometry of the Space-Time Feature; $g(t)$ is serialized and stored with the Space-Time Element to which it belongs. The attributes of the feature at time *t* are represented by $a(t)$. Each Space-Time Element stores the attribute values in a list that corresponds to the attribute name list stored with the Space-Time BLOB. The first element in the attribute value list is the value of the first item in the attribute name list, and so forth. The tuple above demonstrates that the Space-Time Element is capable of capturing the the change in geometry, location, as well as attributes of an object over time.

The Space-Time Element is an abstract class. An abstract class specifies the properties and methods of its subclasses. The abstract class itself cannot be instantiated, but can be subclassed. The subclass inherits all attributes and methods from the super class. In addition, it may have its own properties and methods. In this Space-Time GIS design, the Space Time Element has two subclasses: Space-Time Point and Space-Time Interval. In addition to the properties inherited from the Space-Time Element, a Space-Time Point has one additional property, DateTime, which records the time of its existence. A Space-Time Interval is also a subclass of the Space-Time Element. It has two additional fields, StartDateTime and EndDateTime, which specify the period during which a Space-Time Element exists. After instantiating, a Space-Time Point or a Space-Time Interval will be pushed into the container, that is, the Space-Time BLOB of the Space-Time Feature. A Space Time BLOB can hold very large number of Space-Time Elements, which is the number of times the feature changes its status.

This Space-Time GIS design is event-based. We would not create a new Space-Time Element unless a change occurs to the feature, such as variation on its geometry or attributes. Therefore, each Space-Time Element is associated with one Space-Time Event. A Space-Time Event indicates the reason for creating a new Space-Time Element for the Space-Time Feature. For example, in a historical province map, the boundary of a province might be enlarged because of the administrative division adjustment. Thus, this province would have a new Space-Time Element with a new geometry and attributes related to the administrative division adjustment.

3.3 Creation of Space-Time Object

This section describes how to create the Space-Time BLOB based upon the space-time GIS design introduced in the last section. There are two major components in the creation of the Space-Time BLOB: one is creating the Space-Time BLOB using the available snapshots, and the other is to incorporate the latest data into the created Space-Time BLOB. The methods applied to generate the Space-Time BLOB for snapshots with entity identity and snapshots without entity identity are internally different. These methods are treated separately as follows.

3.3.1 Creation of Space-Time BLOB for snapshots with entity identity

This is the most common situation that might be encountered when creating Space-Time BLOB. The historical data are organized in snapshots, and each object has unique existence. A typical example is the historical state maps of the United States if one were to attempt to organize all the state objects that change over time into Space-Time BLOBs in the Space-Time Feature Class.

The key consideration in creating the Space-Time BLOB is the identity (ID) of the object. This ID is not the OID or FID that is uniquely generated by the system and varies by layers. This ID should be consistent with the object despite its spatial or attribute changes. Thus, one object can be extracted from different snapshots, whether it has changed or not. For instance, in the state maps of the United States, the state name serves as an ID which does not change over time. Although the boundary of Tennessee might be different in 1850 than in 2010, the name "Tennessee" remains consistent.

Since the state of each object at different snapshots should be recorded, it is necessary to cycle through every object in every snapshot. The process demands two strands of data: one representing the loop of snapshots, and one representing the loop of objects in each snapshot (Figure 3.2). The basic idea is to create a Space-Time BLOB for each object with entity identity, no matter how long it existed. The object that exists in only one snapshot would have one Space-Time BLOB, while the object that exists in every snapshot would also have one Space-Time BLOB. The difference exists in the content of the Space-Time BLOB. After getting a snapshot feature class, it is possible to parse the time information of this snapshot, which can be extracted easily by analyzing the name of the feature class.

The next step is to cycle through each object with entity identity in the current snapshot and then to update its Space-Time BLOB. The procedure would create a Space-Time BLOB for an object when processing it for the first time. Returning to the example of creating a Space-Time BLOB for the US state map, when handling the first historical snapshot, no Tennessee Space-Time BLOB has been created since it is the first time to process a Tennessee object. So a Space-Time BLOB for Tennessee will be created at the first step. After that, when getting the Tennessee object in the next snapshot, the Space-Time BLOB of Tennessee will be updated based upon the spatio-temporal information of

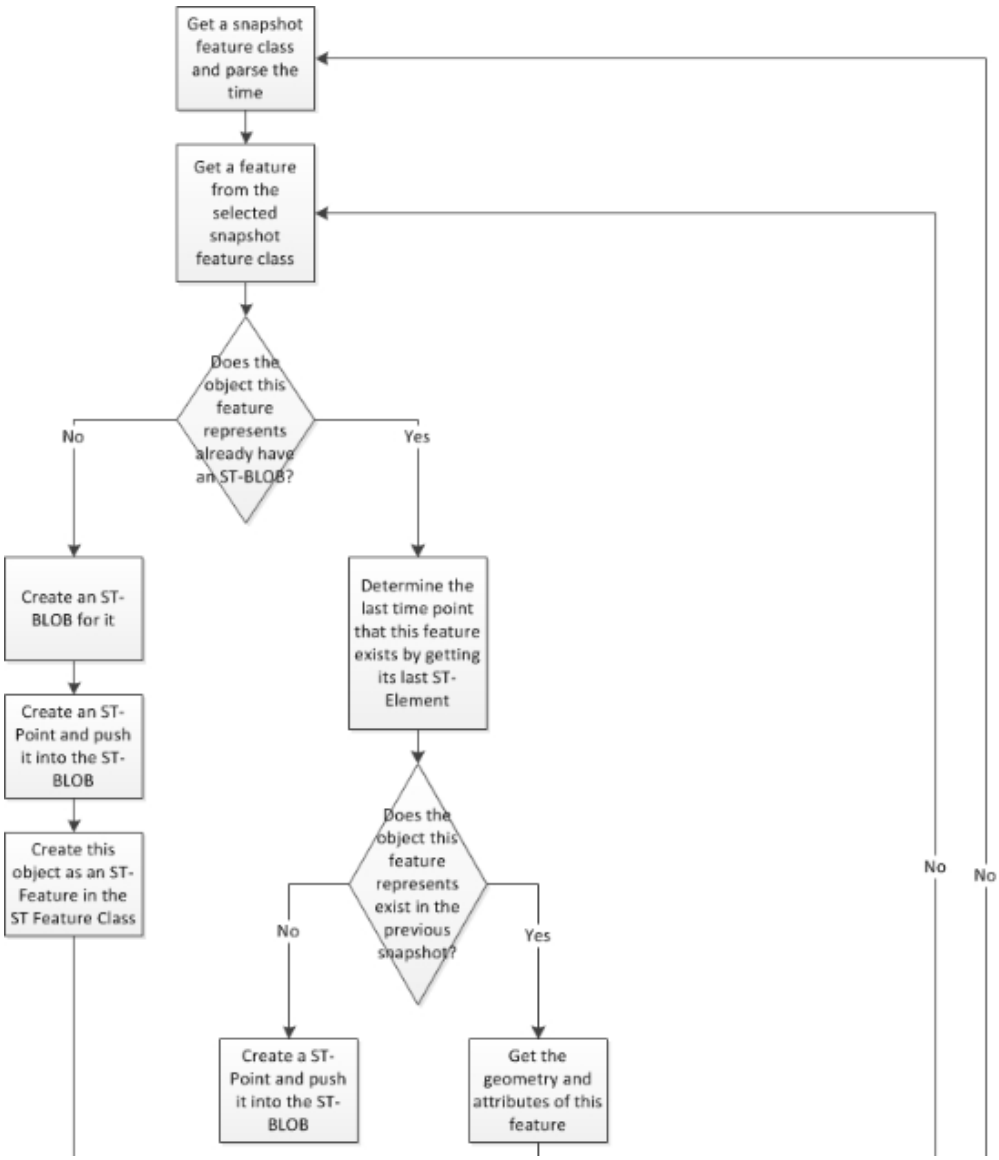


Figure 3.2 The flowchart of creating Space-Time BLOBs for snapshots with entity identity

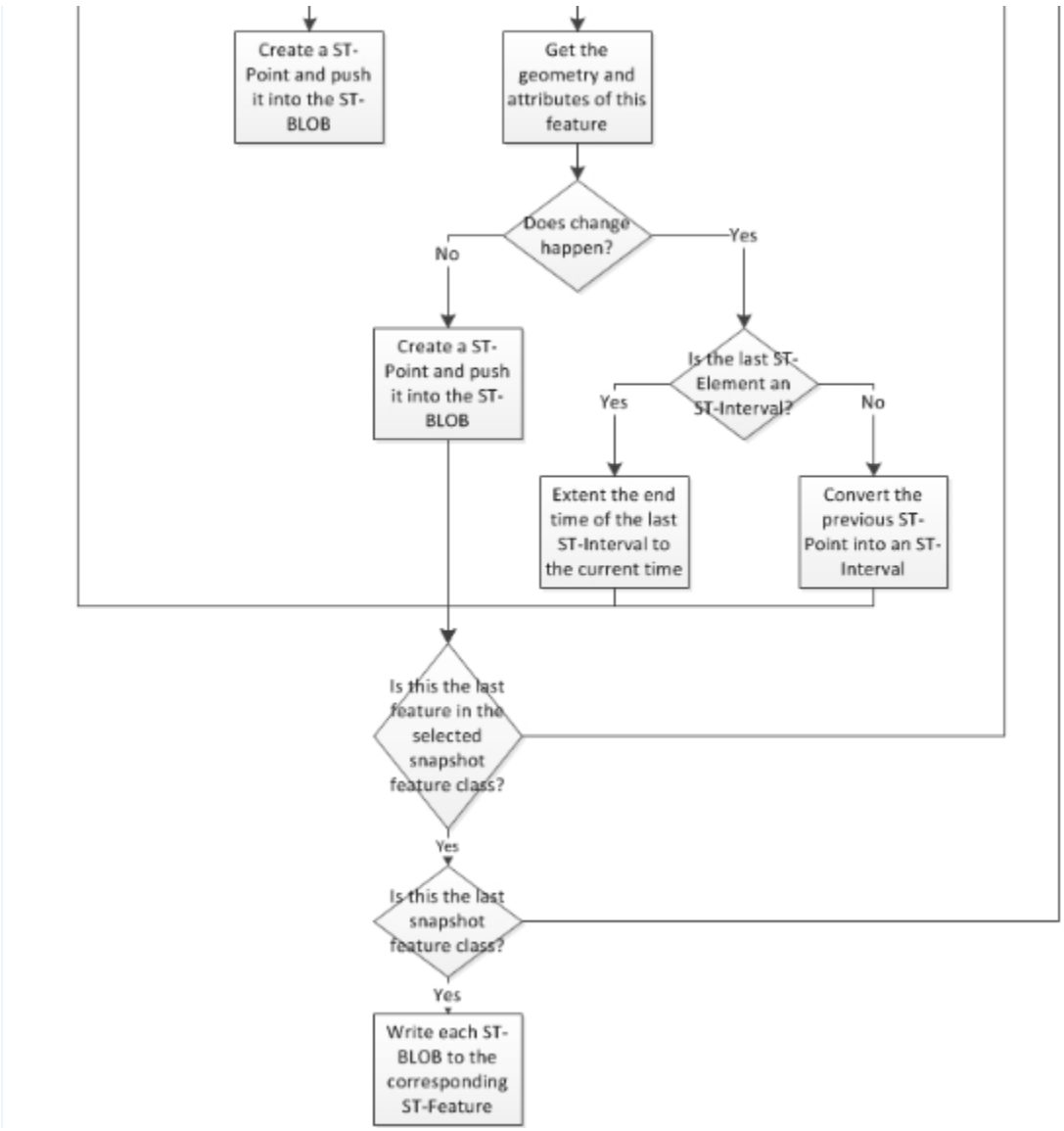


Figure 3.2 The flowchart of creating Space-Time BLOBs for snapshots with entity identity (cont.)

the Tennessee object. The procedure would create a Space-Time BLOB for each object in the first snapshot feature class since it must be the first time an object with entity identity exists. However, whether an object already has a Space-Time BLOB should be determined when getting an object from other snapshots feature class due to the possibility that some objects do not exist in the first snapshot.

The next step is to cycle through each object with entity identity in the current snapshot and then to update its Space-Time BLOB. The procedure would create a Space-Time BLOB for an object when processing it for the first time. Returning to the example of creating a Space-Time BLOB for the US state map, when handling the first historical snapshot, no Tennessee Space-Time BLOB has been created since it is the first time to

process a Tennessee object. So a Space-Time BLOB for Tennessee will be created at the first step. After that, when getting the Tennessee object in the next snapshot, the Space-Time BLOB of Tennessee will be updated based upon the spatio-temporal information of the Tennessee object. The procedure would create a Space-Time BLOB for each object in the first snapshot feature class since it must be the first time an object with entity identity exists. However, whether an object already has a Space-Time BLOB should be determined when getting an object from other snapshots feature class due to the possibility that some objects do not exist in the first snapshot.

When generating a Space-Time BLOB, the procedure creates an attribute name list as a property of the Space-Time BLOB. Therefore, the attribute values at different historical periods can be recorded. The attribute name list is identical to the field names in each snapshot. For example, the state of Tennessee might have Population and GDP fields. Population and GDP are considered to be attribute names and will be captured to form a list stored in Tennessee Space-Time BLOB. In some cases, when later snapshots might hold more attributes than the earlier ones additional processes would be taken into consideration. If a snapshot does not maintain an attribute that other snapshot does, the attribute values of objects in this snapshot would be recorded in the Space-Time Elements as NULL.

A Space-Time Point will be created after generating the Space-Time BLOB. The time information which is extracted from the layer name is stored as a property of the Space-Time Point. The procedure accesses the object's geometry and then serializes it with the Space-Time Point. In addition, the Space-Time Point has a property referred to as attribute value list, which corresponds to the attribute name list in the Space-Time BLOB. The values can be obtained from the object in the snapshot feature class that the procedure is cycling through.

If an object with entity identity already has its Space-Time BLOB, which means it appears at least once in the previous snapshots, it is not necessary to generate a Space-Time BLOB for this object again. Instead, the task is to determine whether this object exists in the previous snapshot. "No" indicates a gap between the last time this object existed and the current time this snapshot feature class represents. Thus, this object must disappear in one or more snapshot(s). For example, a person who appears in the study area yesterday might disappear today and appear again tomorrow. Therefore, we can directly create a Space-Time Point for this object and then push this Space-Time Point into its Space-Time BLOB.

If this object with entity identity exists in the previous snapshot feature class, it means that this object does not disappear during the time between the current snapshot and the previous one, although there is no way to ascertain what happens in between given the specific temporal resolution. The process becomes more complex when determining if any change occurs to this object since the previous snapshot feature class, including spatial and attribute changes.

To determine spatial change appears to be simple, while it actually is not. Features in snapshots are always digitized by hand. An object in a later snapshot feature class is usually not copied and pasted from an earlier snapshot feature class, even if in fact they should be the same. The digitizing process inevitably introduces inconsistencies since it is impossible to preserve the geometry 100%, even if it does not change in the real world. In this study, less than 1% difference in area is considered to be no change. Although not perfect, this method is simple and proved to be effective for the Chinese historical provinces scenario mentioned in Chapter 1. Meanwhile, any inconsistencies in specified key attributes indicate that change happens.

Detected changes on either geometry or attributes imply that a particular object has started a new status. Therefore, we need a new Space-Time Point with the latest spatio-temporal information, which is then pushed into the Space-Time BLOB. On the other hand, if no spatio-temporal change is detected, then that object has extended its states; its geometry and attributes remain the same since the previous snapshot feature class. Such continuation implies a Space-Time Interval; however, two sub-situations need to be taken into consideration. On the one hand, if the previous Space-Time Element is a Space-Time Point, the procedure converts it into a Space-Time Interval. The start time of this Space-Time Interval is the time of the converted Space-Time Point, while the end time is the time of the current snapshot feature class. On the other hand, if the previous Space-Time Element is already a Space-Time Interval, the procedure will simply assign the time of the current snapshot class to its end time.

After generating a Space-Time BLOB for each object with entity identity, each object is represented as a Space-Time Feature in the Space-Time Feature Class. It appears as its initial geometry, while its Space-Time BLOB contains all the spatio-temporal information. Every Space-Time BLOB would be written into the corresponding Space-Time Feature's STBLOB field. Thus, the procedure is complete.

3.3.2 Incorporation of new data into the Space-Time BLOB of object with entity identity

When the process completes all the snapshot classes, each Space-Time Feature would have a Space-Time BLOB. At that point, people do not need to capture a snapshot of every particular period any more. Instead, the data are managed based on the event. An event refers to the spatial or attributes change of the entity, which triggers the procedure that updates or creates the Space-Time BLOB to reflect changes. Compared with creating Space-Time BLOB for snapshot feature class with entity, the difference in incorporating new data is that this procedure handles only the updated or created objects, so it does not cycle through any snapshots feature class. If the object is newly created, the process will create a Space-Time BLOB with a Space-Time Point for it. If the object is updated, it is not necessary to determine whether it is different than its previous states—it must be different—so a Space-Time Point can be created and and pushed into the object's Space-Time BLOB. Finally, the created or updated Space-Time BLOB will

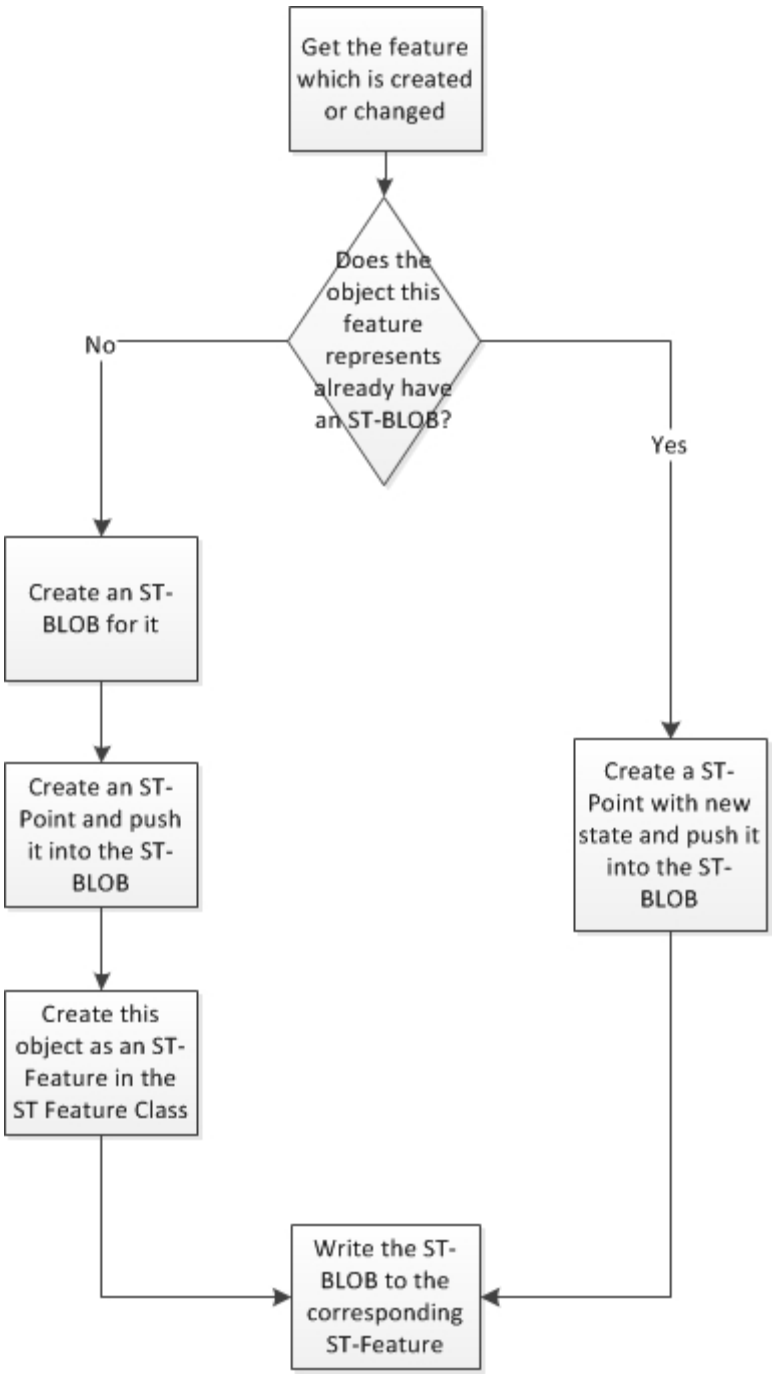


Figure 3.3 The flowchart of incorporating new data into Space-Time BLOBs with entity identity

be re-written to the corresponding Space-Time Feature in the Space-Time Feature Class (Figure 3.3).

3.3.3 Creation of Space-Time BLOB for snapshots without entity identity

Sometimes an object in the snapshot feature class does not have an entity identity in the real world. Instead, it represents a status of a location of an area. In other words, it does not have independent unique existence. The land use and soil pH value maps are typical cases in this category.

These phenomena are also usually recorded as snapshots. There is no ID to identify each object in the snapshot without entity identity. When creating the Space-Time BLOB, the task is to record how each spatial unit changes over time.

In this study, we applied the space-time composite model to record the spatio-temporal information for each so-called greatest common unit derived by overlaying all the snapshots and then breaking all the features apart. The spatio-temporal information of the whole study area can be represented by all of the greatest common units. For example, to explain how this method works: At T_1, we have polygon A, B and C. At T_2, we have polygon D, E and F (F is the same as C). Overlaying these two layers and conducting an "intersect" and a "difference" operations generates the greatest common units. The resulting layer consists of the eight greatest common units (Figure 3.4). A generated composite table explains whether a greatest common unit exists in each snapshot layer (Table 3.1).

To create the Space-Time BLOB for the snapshot feature class without entity identity, all historical objects are overlaid and intersected with one another to generate the greatest common units in the first step. These features are stored in the Space-Time Feature Class.

There are also two loops in this procedure: one for the greatest common units in the Space-Time Feature Class, and the other for the snapshot feature classes without entity identity. It appears that, despite some variation, it is very similar to the flowchart of Space-Time BLOB creation for snapshot feature class with entity identity (Figure 3.2).

The research process created the Space-Time BLOBs for each greatest common unit and then populated the Space-Time BLOB by determining its existence and status in each snapshot feature class. This Space-Time BLOB serves as a container for the Space-Time Element (i.e., Space-Time Point or Space-Time Interval) that records the spatio-temporal information of each greatest common unit. In the meantime, an attribute name list is initiated; each Space-Time Element contains an attribute value list which corresponds to the attribute name list in the Space-Time BLOB.

As described above, after initiating a Space-Time BLOB for one greatest common unit, the procedure uses a loop to check its status in each snapshot feature class. The time information is extracted from the name of the snapshot feature class which also obeys the nomenclature as the snapshot with entity identity. On one hand, if a greatest common unit does not exist in a snapshot feature class, it means that this greatest common unit does not have any Space-Time Element at the time when this snapshot is taken. Under

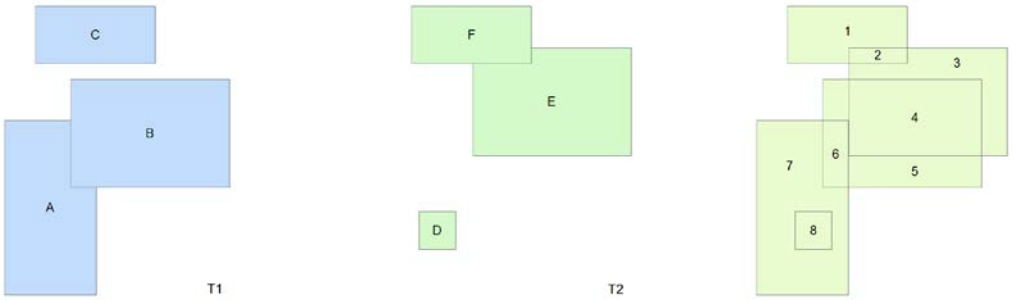


Figure 3.4 An example of generating greatest common units

Table 3.1 The generated table of greatest common units

	1	2	3	4	5	6	7	8
T1	C	C	-	B	B	A/B	A	A
T2	F	E/F	E	E	-	-	-	D

this circumstance, the procedure would do nothing and then runs into the next snapshot feature class. On the other hand, its appearance in a snapshot feature class indicates the greatest common unit either has a Space-Time Point or a Space-Time Interval that covers the time of the snapshot feature class. The attribute values of the greatest common unit at this time would be extracted from the object in the snapshot feature class this greatest common unit overlaps.

The procedure determines whether this greatest common unit exists in the previous snapshot feature class. A negative answer means this greatest common unit disappears for at least one snapshot, a Space-Time Point will be generated for it. The created Space-Time Point is then pushed into the Space-Time BLOB of this greatest common unit. A positive answer indicates this greatest common unit does not disappear during the time between the current snapshot and the previous one (although, given the specific temporal resolution, there is no way to know what happens in between). In this case, the question becomes whether any change occurs on this greatest common unit since the previous snapshot feature class.

For the object without entity identity, it is not necessary to detect the spatial change since all the objects in different snapshots have already been intersected into greatest common units. These greatest common units cover the whole study area, and their geometries are consistent. Therefore, the procedure does not need to compare two consecutive geometries and then determine whether they are the same.

It is still necessary to compare attributes between the previous Space-Time Element and the current feature this greatest common unit overlaps. Any variations imply that this greatest common unit has a new status, so the procedure will create a new Space-Time Point with the latest attributes. This Space-Time Point is then pushed into the Space-Time BLOB of the greatest common unit. However, if the snapshot feature classes hold

too many attributes, it may be useful to specify the key attributes. Only variations that happen to these key attributes could be considered as changes.

No attribute change indicates that this greatest common unit extends its status to the current cycle of snapshot; such continuation implies a Space Time Interval. Also, if the previous Space-Time Element is a Space-Time Point, the procedure will convert it into a Space-Time Interval. The start time of this Space-Time Interval is the time of the converted Space-Time Point, while the end time is the time of the feature class of the current cycle. On the other hand, if the previous Space-Time Element is already a Space-Time Interval, the procedure will simply assign the time of the feature class in the current cycle to its end time.

The Space-Time BLOB creation for a greatest common unit is finalized after cycling through all snapshot feature classes for this feature. The process will run into the next greatest common unit, and then all snapshots feature classes would be cycled through again, until every greatest common unit is assigned a Space-Time BLOBs.

3.3.4 Incorporation of new data into the Space-Time BLOB of object without entity identity

Once the above procedure is accomplished, there must be a procedure for handling new data. For the objects with entity identity data, we could update the Space-Time BLOB of the changed object without taking snapshots for all objects. Nevertheless, for the objects without entity identity, such as the land use data, the update method is still based on new images.

Maintaining the created Space-Time Feature Class is cumbersome. When obtaining a new snapshot layer, this layer would be overlaid with the Space-Time Feature Class, which consists of all greatest common units, to generate a new Space-Time Feature Class with new greatest common units. The number of features in the new Space-Time Feature Class could be equal to the number of features in the previous Space-Time Feature Class when the overlay operation does not generate more greatest common units, or more than the number of features in the previous Space-Time Feature Class when the overlay operation generate more greatest common units. This process is cumbersome but inevitable. It is the major weakness of the space-time composite approach: features need to be overlaid and intersected again when data are updated to ensure that all the geometries are the "smallest." Possible improvement of the space-time composite is not within the scope of this study. In the future it might be feasible to construct a better method for dealing with objects without entity identity.

A procedure cycles through each latest, greatest common unit and compares it with the greatest common units in the previous Space-Time Feature Class. In some cases, they are completely the same, and it is a matter of simply transferring the Space-Time BLOB from its ancestor, later determining whether change applies to attributes. If the attributes are still the same, the procedure converts the latest Space-Time Point into Space-Time

Interval (if the last ST-Element is a Space-Time Point) or extends the latest Space-Time Interval (if the last Space-Time Element is a Space-Time Interval). If there are any changes in attributes, the next step creates a Space-Time Point and pushes it into the Space-Time BLOB, perhaps again only considering the key attributes to reduce unnecessary data redundancy. In other cases, a previous greatest common unit might contain more than one current greatest common units, which means transferring the Space-Time BLOB from the previous greatest common unit, and then processing the new Space-Time Element in the same way as the first case. Sometimes, a new greatest common unit does not overlap with any feature in the previous Space-Time Feature Class, thus this greatest common unit occupies a new area. That condition would initiate a Space-Time BLOB with a Space-Time Point that contains the attribute information.

3.4 Storage of the Space-Time BLOB

The previous section discussed how to create Space-Time BLOBs for object with/without entity identity by noting "create a Space-Time BLOB for an entity" or "create a Space-Time BLOB for a greatest common unit," without illustrating "how to" store a Space-Time BLOB with a Space-Time Feature or a greatest common unit. When creating an object, the program allocates some space in memory for this object. This space would be released after the program shuts down. However, storing all Space-Time BLOBs in the memory is unrealistic. On the one hand, the number of objects in the study area in the world might be tremendous, perhaps as over a million. This imposes a great deal of overhead in the system. Keeping and managing such a number of Space-Time BLOBs in memory also raises questions about efficiency. On the other hand, we must keep these Space-Time BLOBs in other storage formats such as a hard drive. When closing the application, these Space-Time BLOBs should remain in that drive for future use. Therefore, the problem is how to transfer a Space-Time BLOB to a format that can be stored physically and permanently in the disk and transferred back into Space-Time BLOB in order to support spatio-temporal query.

In computer science, a mechanism used to convert a data structure or an object into a physical format is known as *serialization* (Figure 3.5). Conversely, the opposite direction of this operation—*deserialization*—is used to "resurrect" the serialized Space-Time BLOB so that it can be manipulated by the application (Palmer and Barker 2008). Again, a BLOB type field in the Space-Time Feature Class stores the spatio-temporal information, thus explaining the way on the Space-Time BLOB obtained its name.

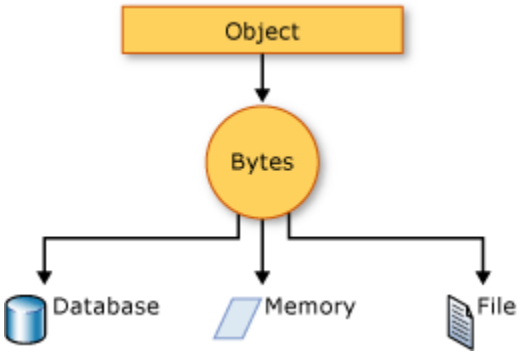


Figure 3.5 The process of serialization

This chapter investigates the limitations of snapshot model in detail and introduces the object-oriented view of space and time applied in this study. The design of the model is discussed, as well as the workflows to import historical data and to incorporate new data into this model. Finally, the mechanism used to store the Space-Time BLOB is presented. The next chapter discusses the prototype system which implemented the ideas in Chapter 3, using Chinese historical province maps and historical land use maps on the campus of University of Tennessee as scenarios.

CHAPTER 4 A SPACE-TIME GIS PROTOTYPE SYSTEM

The prototype system described in this chapter illustrates the functionalities of the proposed Space-Time GIS design. The model design and prototype system implementation processes operated simultaneously. The model design provided a guideline to the prototype system, while the problems found in implementation helped revise and improve the prototype system. This chapter describes how the proposed Space-Time GIS design is implemented using a commercial GIS developer kit, demonstrating functions such as Space-Time BLOB creation and Spatio-temporal query. Two datasets—historical Chinese provinces and historical land use on the campus of University of Tennessee (UT)—were chosen to test the snapshots separately with/without entity identity scenarios. The following establishes why the proposed space-time GIS in this study is superior to the traditional snapshot model.

4.1 The Prototype System and the Implementation Platform

In the prototype system, users were able to import the spatio-temporal data and store them against the suggested space-time GIS data model. This process is fully automatic as long as users follow the nomenclature and organize the historical snapshots in a specified way before using the tool. The reason an automatic process is necessary is that it is not uncommon to have tens of thousands of historical snapshots; creating Space-Time BLOBs manually is not realistic. After creating the Space-Time BLOBs, the spatio-temporal information of the historical objects, such as the geometry and attribute of a feature at a given time would be observable. Users can perform six types of spatio-temporal queries in the created Space-Time Feature Class. The data review and data query process was object-based, wherein each Space-Time BLOB for historical object contained all the spatio-temporal information for this object. In other words, by applying the proposed space-time GIS data model, it is possible to take advantage of this object-oriented spatio-temporal data storage method. Data review and data query did not require cycling through each snapshot, which is inefficient and time-consuming.

Two datasets provided scenarios: historical Chinese provinces provided data for the snapshot with entity identity scenario, and the historical land use map on the campus of University of Tennessee (UT) provided data without identity.

A province is a typical example of object with entity identity. Each province has a specific identity over time with its name as its ID. The dataset used here included five historical layers of 1953, 1964, 1982, 1990, and 2000 (Figure 4.1). Over the course of these years, many administrative provincial boundaries were adjusted several times. Some provinces were divided and merged with other provinces; for example, many counties in Hebei Province became parts of Beijing in 1950s; Chongqing became independent from Sichuan in 1997; Tianjin was a city of Hebei Province in 1958 and was

1953	1964
	
1982	1990
	
2000	
	

Figure 4.1 Historical Chinese province maps

designated as a municipality in 1966. In addition, the administrative boundaries of several provinces had not changed since 1953. This is a useful scenario for testing the workflow of Space-Time BLOB creation for snapshot with entity identity class. Although maps from 1964 to 2000 look very similar, slight spatial variations do exist.

A land use parcel can be treated as an object without entity identity since it does not keep an identity through time. In this study, we obtained the Knox County historical land use data from Knox County Metropolitan Planning Commission (MPC). The historical land use parcels on the UT campus in 1998, 2003 and 2008 are extracted as the scenario. There were 663, 672, 581 land use parcels in this area in these three years, respectively. The extents and types of land use parcels went through certain changes during this period. Figure 4.2 illustrates such difference of land use parcel extents in a small region on the UT campus.

The following were used to implement this prototype: ArcGIS Engine 9.3.1 as the

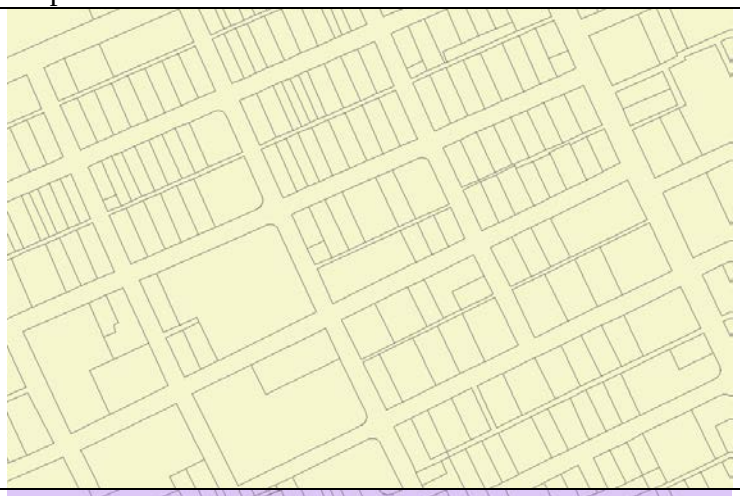
Year	Map
1998	
2003	
2008	

Figure 4.2 Historical land use parcels on UT campus

developer kit and Visual Studio 2008 as the integrated development environment (IDE); C# 2008 was selected as the programming language; ArcGIS Engine as a set of the core ArcObjects components with several controls for developers, such as MapControl, TOCControl, ToolbarControl, and so on. Therefore, it was possible to take full advantage of the existing comprehensive and powerful ArcGIS functionalities, from data storage and access in Geodatabase geometric operations, to data representation and visualization, which allowed research to focus on the implementation of the Space-Time Classes discussed in Chapter 3. Another potential benefit of using the ArcGIS Engine is that users in the future do not need to install the full ArcGIS package to run the prototype, only the ArcGIS Engine Runtime is necessary.

In March of 2011, migrating the prototype system to ArcGIS Enging 10.0 and Visual Studio 2010 ensured that the libraries were the latest version. .Net Framework 3.5 SP1 was the framework compiled against, rather than the latest .Net Framework 4.0, since ArcGIS Engine 10.0 class library does not yet support .Net Framework 4.0.

4.2 System Overview

The system interface design (Figure 4.3) displays a Table of Content (TOC) control, which shows all historical snapshots and the Space-Time Feature Class, on the left side. The organization of historical snapshots in Geodatabase before processing is described in 4.3.1. On the right top, a toolbar control contains the basic map tools, such as Add Data, ZoomIn and ZoomOut, Full Extent, and so on. The map control in the middle shows the map of selected layers. At the bottom left is a list of the OBJECTID and ID of each Space-Time Feature in the Space-Time Feature Class; the ID is the key that maintains the identity through time of an object with entity identity. The ID is assigned arbitrarily for object without entity identity. A tab control at the central bottom has three tabs. The first, ST-BLOB, shows the Space-Time BLOB of the selected Space-Time Feature. This tab also contains a Create ST-BLOB button. The Change Type label in this tab is used to indicate what kind of spatio-temporal changes the selected Space-Time element undergoes compared with its previous Space-Time element. Geometry represents change that occurs in geometry, attribute represents change in attributes, while new indicates that the selected Space-Time Element does not exist in the previous time point. The second tab, ST-Query, allows users to perform six types of spatio-temporal queries on the Space-Time Feature Class. The third tab, Space-Time Analysis, is for future development (this study does not include Space-Time Analysis capabilities). The functions of the prototype are introduced in the remainder of this chapter.

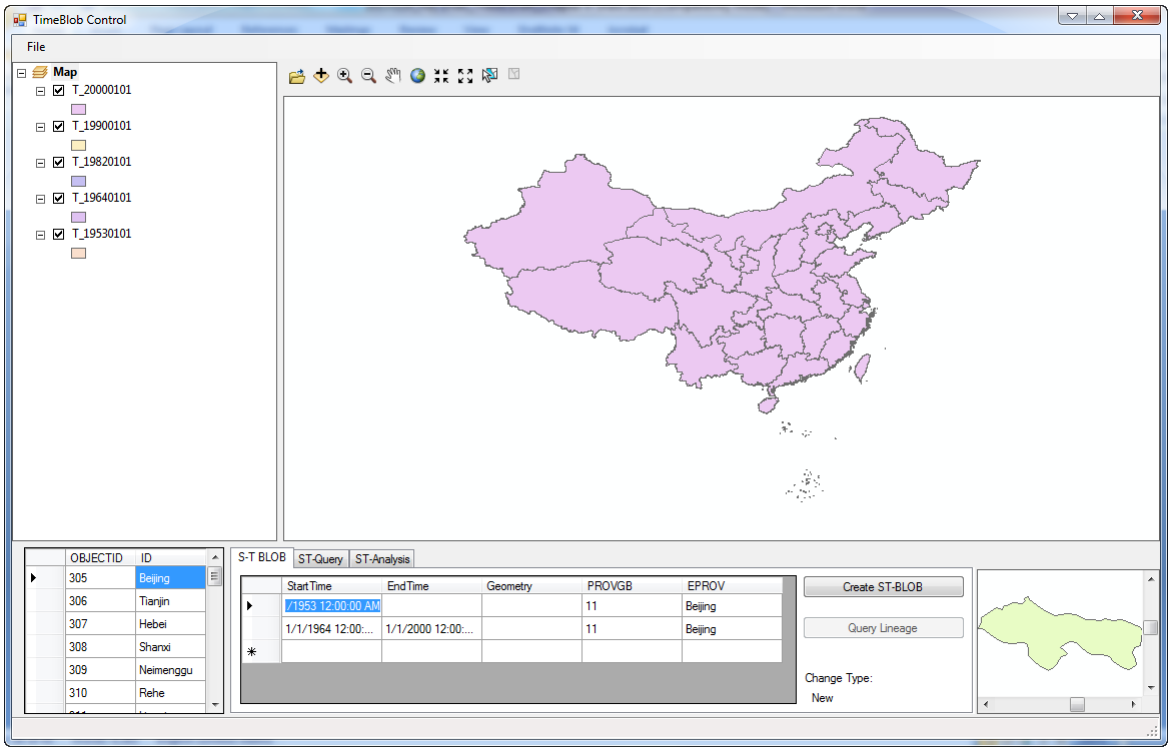


Figure 4.3 The interface of the prototype system

4.3 Tools for Space-Time BLOB Creation in Geodatabase

In order to create Space-Time BLOBs for historical snapshots, corresponding tools were developed. These tools generated the Space-Time Feature Class with Space-Time Features, each of which comes with a Space-Time BLOB. This section describes how the historical data must be organized in Geodatabase before processing. The tools for creating Space-Time BLOB for snapshots with and without entity identity, respectively, are then introduced.

4.3.1 Data Preparation before processing

The prerequisite for creating Space-Time BLOB is to organize the historical data properly in the Geodatabase so that the program is able to read the data from the specific dataset and process them correctly. This step is important because different historical datasets have different nomenclature. The prototype system requires a unified nomenclature on these datasets so that the temporal information of the snapshots can be extracted correctly.

First, the prototype system manages the historical datasets in personal Geodatabase, which is based upon Microsoft Access. Since the historical data may come in different formats, such as shapefile or coverage, they must be transformed into Geodatabase. Users could utilize the import function in ArcCatalog to handle multiple historical

snapshots at one time. Each snapshot would be transformed into a feature class in the Geodatabase; these feature classes are managed under a feature dataset, which is a collection of spatially or thematically related feature classes sharing a common coordinate system (Zeiler 2010). This is like a folder in the operating system within which files can be integrated and managed.

The name of the historical snapshot is required to follow the nomenclature. It starts with a T and an underline (_), followed by the date time string. The feature class name "T_19861104" means the snapshot was taken on November 4th, 1986. More often than not, users do not have detailed information about the date when the historical datasets are created. In this case, the only thing possible is to specify an estimated year and assign 0101 for month and day. In other cases, users might have snapshots that were taken every hour, or even every minute. They could extend the nomenclature to include more detailed time information. For example, "T_19861104113005" means this snapshot was taken at 11:30:05 on November 4th, 1986.

The organization of the Space-Time Geodatabase for the Chinese historical province scenario utilizes personal Geodatabase (Figure 4.4). File Geodatabase is preferred if users have very large amount of historical data. Three feature datasets are necessary in this prototype system: the first is the feature dataset that organizes the historical snapshots, which data might be transformed from the formats other than feature class. The second feature dataset, QueryResult, stores the generated result of the spatio-temporal query, whose feature classes are temporary and will be removed before the next spatio-temporal query. The third feature dataset: Temp, has one feature class to store and visualize the geometry of the selected Space-Time Element.

4.3.2 Tools to Create Space-Time BLOB for snapshots with entity identity

Once the data are ready, the tool can create Space-Time BLOBs for historical data. As mentioned above, the internal mechanisms for creating Space-Time BLOBs for snapshots with and without entity identity are different. This section covers Space-Time BLOBs for snapshots with entity identity.

Clicking the Create ST-BLOB button (Figure 4.5) would begin to process each historical snapshot. Section 3.3.1 details the workflow. The time needed to finish this ST-BLOB creation varies with respect to the number of historical data layers. It takes longer to process large historical datasets.

The generated Space-Time Feature Class is added to the map when the creation process is finished. This feature class consists of all the objects that ever appeared in history, with each of them appearing in its initial state. The spatio-temporal information is integrated into corresponding Space-Time BLOBs of each historical feature. At this point, the list of the IDs of Space-Time Features has been populated in the Historical Feature List in the bottom left, along with the OBJECTIDs.

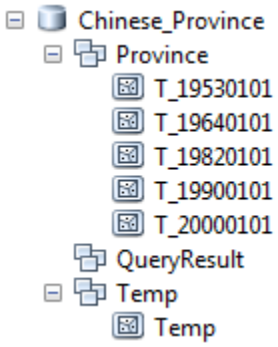


Figure 4.4 The organization of the space-time Geodatabase



Figure 4.5 The tab of Space-Time BLOB creation

Right after the user clicks a record in the Historical Feature list, its corresponding Space-Time BLOB will be shown in a table in the ST-BLOB tab. Each row of in the table of Space-Time BLOB represents a Space-Time Element. This may be a Space-Time Point (when the field EndTime is left blank) or a Space-Time Interval (where both StartTime and EndTime fields have values). If users click the geometry cell of a Space-Time Element, the geometry of the selected Space-Time Element will be shown in the preview map. The corresponding Space-Time Feature in the Space-Time Feature Class is highlighted in the map simultaneously. The attribute information stored with each Space-Time Element is also displayed in this table. This design is makes it possible for users to see how an object changes over time, such as the number of changes (the number of Space-Time Elements) and the time of each change. This contrasts with the *snapshot model* from which it is not easy to extract such information about an object.

For example, if Beijing is chosen from the list of Space-Time Feature, its Space-Time BLOB appears in a table under the S-T BLOB tab. Beijing has two Space-Time Elements: one is a Space-Time Point on 1/1/1953 (Figure 4.6), and the other is a Space-Time Interval from 1/1/1964 to 1/1/2000 (Figure 4.7). Thus, Beijing appears in the historical snapshot of 1953; its geometry changed in 1964, which is indicated in the right bottom corner of the Space-Time BLOB. Its status remains consistent to 1/1/2000, meaning it did not change in the intermediate historical snapshot (i.e., 1982 and 1990). This exactly describes the circumstances in Beijing during the 1950s. From late 1952 to

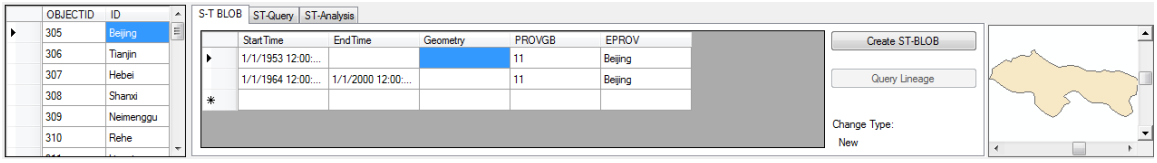


Figure 4.6 Beijing's Space-Time Point in 1953

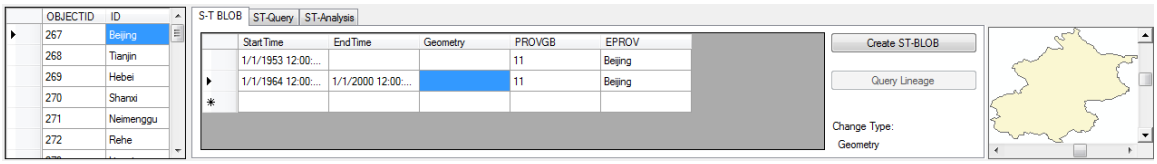


Figure 4.7 Beijing's Space-Time Interval from 1964 to 2000

1960, many counties in Hebei Province located next to Beijing, such as Fangshan County, became part of Beijing City. Therefore, in the model the area of Beijing dramatically changes in shape from Beijing's Space-Time BLOB (1964-2000). Using the snapshot model would require cycling through each snapshot to extract Beijing's geometry and attributes, conducting a series of comparison processes between consecutive Beijing objects, and finally arriving at information similar to that which is directly reflected in Beijing's Space-Time BLOB.

Since Beijing is assigned a Space-Time Point in 1/1/1953 and a Space-Time Interval from 1/1/1964 to 1/1/2000, people might be curious about Beijing's state between 1953 and 1964. The prototype system would return nothing if people query Beijing's state in 1958 due to the fact that people have no way to understand when change occurred between two consecutive snapshots. However, this study still assumes that nothing happens during a Space-Time Interval, while in fact during this interval change could occur on an object which returned to its original state before the end time of the interval. Different view points out that a Space-Time Interval should be created from 1/1/1953 to 1/1/1964, instead of leaving a gap between this period, since these are all we can get from the snapshots. This study insists the viewpoint that changes must occur on Beijing between 1953 and 1964 while changes might occur on it between 1964 and 2000. There is no perfect solution for this debate. It should be noted that when performing spatio-temporal query on historical data, users need to be very careful with the uncertainty exists in the historical snapshots.

Another example involves Chongqing City, which was a major city in Sichuan Province until 1997 when it became a municipality directly under the jurisdiction of the central government (Figure 4.8). Such a municipality is administratively at the same level as a province; as a result, it does not appear in the historical province maps until 2000.

Many provinces did not experience any administrative boundary adjustments after 1953; in those cases, only one Space-Time Element (from 1953 to 2000) would exist in their Space-Time BLOBs. In the proposed Space-Time GIS data model, the Space-Time

BLOB of Fujian Province requires only one Space-Time Element (Figure 4.9). Users would know immediately that Fujian remains the same from 1950 to 2000 without any spatial or attribute changes (i.e., the code and name of Fujian Province did not change during 1953 and 2000). On the contrary, the snapshot model would be unable to provide that information.

Heilongjiang Province is another kind of extreme example, having undergone frequent administrative boundary adjustments from 1953 to 2000. As a result, its shape varies between each historical snapshot, which is why Heilongjiang Province has five Space-Time Points in its Space-Time BLOB (Figure 4.10 to Figure 4.14). Each Space-Time Point stores its status at a specific time point, providing users with a quick overview of at least five changes in the province from 1953 to 2000, "at least" because the snapshot model does not track changes. It is possible that changes happen more than once between two consecutive snapshots.

4.3.3 Tools to Create Space-Time BLOB for snapshots without entity identity

The data organization method for object without entity identity is the same as it is for object with entity identity. All the historical snapshots will be transformed from other formats to feature class in the Geodatabase under a common feature dataset

After the data is prepared, the user clicks the Create ST-BLOB button. The prototype system overlays all the historical snapshots and then intersects them into greatest common units. In the UT land use scenario, 3915 greatest common units were generated. Many of them are sliver polygons introduced by deviations when digitizing the maps of different years. Subsequently, the system is able to create a Space-Time BLOB for each greatest common unit by cycling through historical snapshots (Section 3.3.3). In this study, we did not develop extra process to remove sliver polygons. As a result, each sliver polygon would have a Space-Time BLOB that includes complete spatio-temporal information for this sliver polygon.

Also, a Space-Time Feature Class which includes all greatest common units would be created and added to the map (Figure 4.15). The corresponding Space-Time BLOB is displayed in the ST-BLOB tab after the user selects a greatest common unit in the bottom left list. The Space-Time BLOB table for object without entity identity can be interpreted in the same way as it is for object with entity identity: each row represents a Space-Time Element, either a Space-Time Point (without an end time) or a Space-Time Interval (with an end time). In addition, the attributes of the greatest common unit in each Space-Time Element are displayed in this table. Users can review the geometry of the greatest common unit by clicking the geometry cell in one of the Space-Time Elements.

Figure 4.16 shows a greatest common unit with ObjectID 1853. It has only one Space-Time Element with start time 1998 and end time 2008 which indicates that it was always a multifamily residential land in 1998, 2003 and 2008 given the specified temporal resolution (5 years).

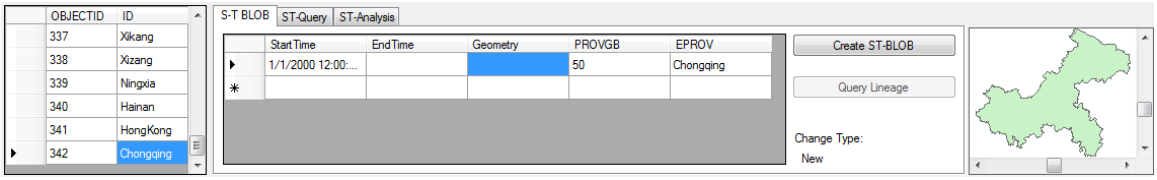


Figure 4.8 Chongqing's Space-Time Point in 2000

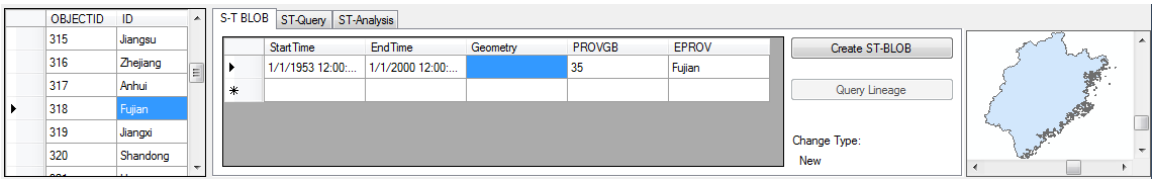


Figure 4.9 Fujian's Space-Time Interval from 1953 to 2000

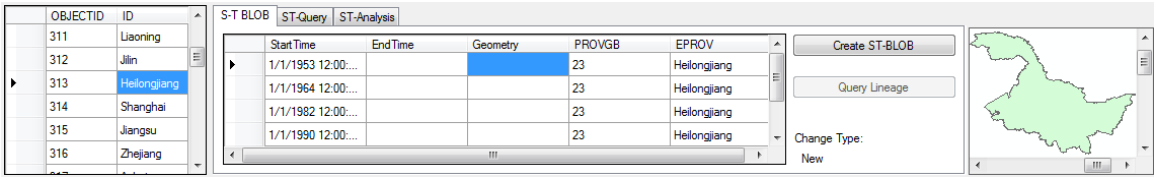


Figure 4.10 Heilongjiang's Space-Time Point in 1953

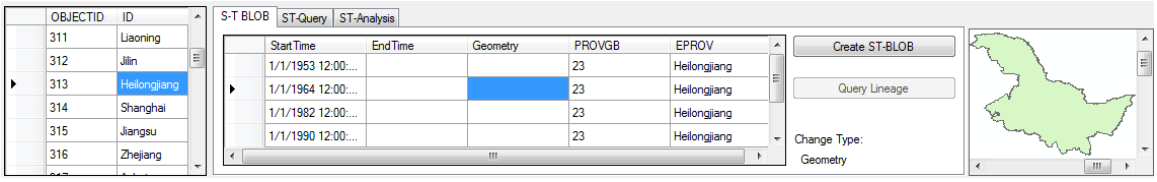


Figure 4.11 Heilongjiang's Space-Time Point in 1964

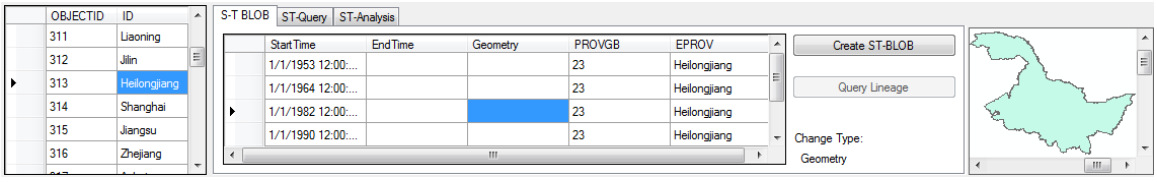


Figure 4.12 Heilongjiang's Space-Time Point in 1982

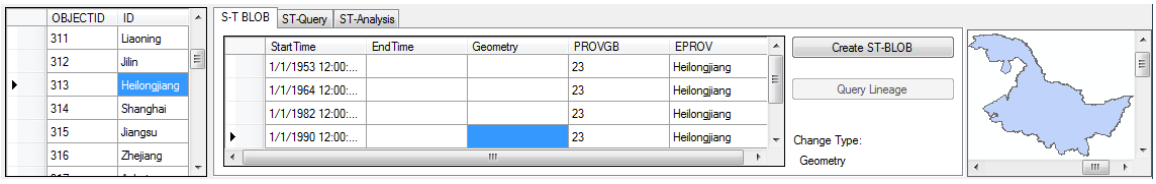


Figure 4.13 Heilongjiang's Space-Time Point in 1990

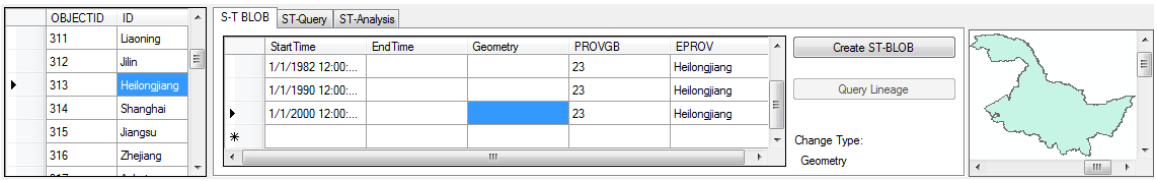


Figure 4.14 Heilongjiang's Space-Time Point in 2000



Figure 4.15 The generated greatest common units

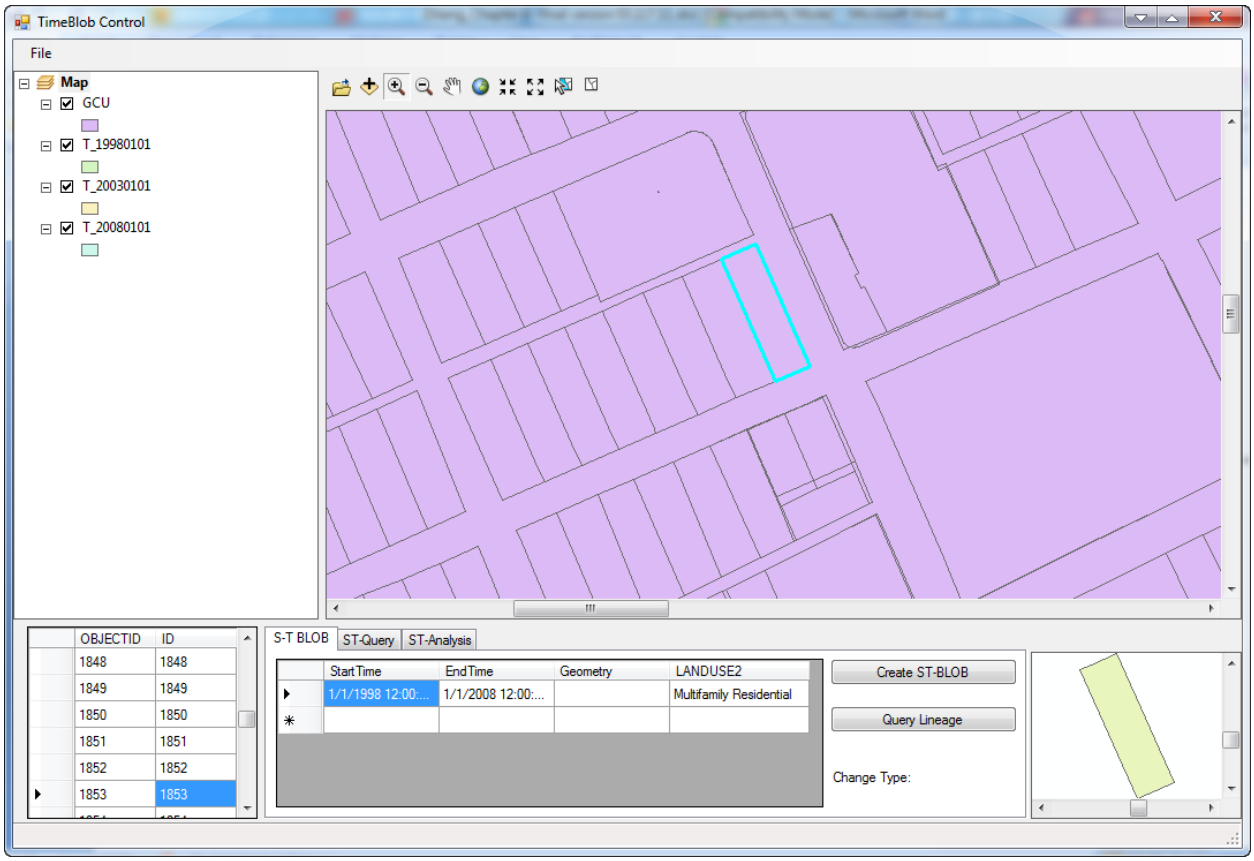


Figure 4.16 An example of a greatest common unit with a Space-Time Interval from 1998 to 2008

The land use parcel with ObjectID 1878 has two Space-Time Elements. One is a Space-Time Interval from 1998 to 2003. The other is a Space-Time Point in 2008 (Figure 4.17). Therefore, we know that land parcel 1878 was an agricultural/forestry/vacant land in 1998 and 2003. It became a transportation/communication/utilities parcel in 2008.

In some cases, users might be curious about the lineage of the greatest common unit so as to track the changes in each greatest common unit; they want to know which object this greatest common unit belonged to at a given time point. This information is processed and stored with each Space-Time BLOB. Users could obtain this information by clicking the Query Lineage button (Figure 4.18). For example, the greatest common unit with ObjectID 2585 belonged to the feature with ObjectID 510 in 1998 and the feature with ObjectID 272 in 2003 when it was a wholesale parcel. It became a multifamily residential in 2008 and belonged to the feature with ObjectID 276.

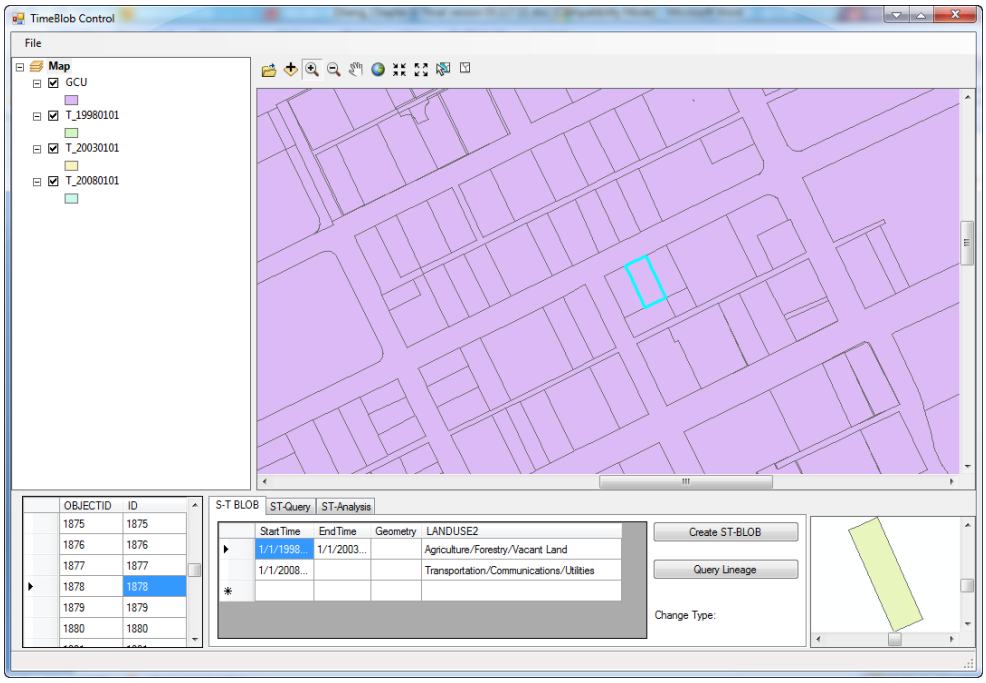


Figure 4.17 Another example of a greatest common unit with a Space-Time Interval from 1998 to 2003 and a Space-Time Point in 2008

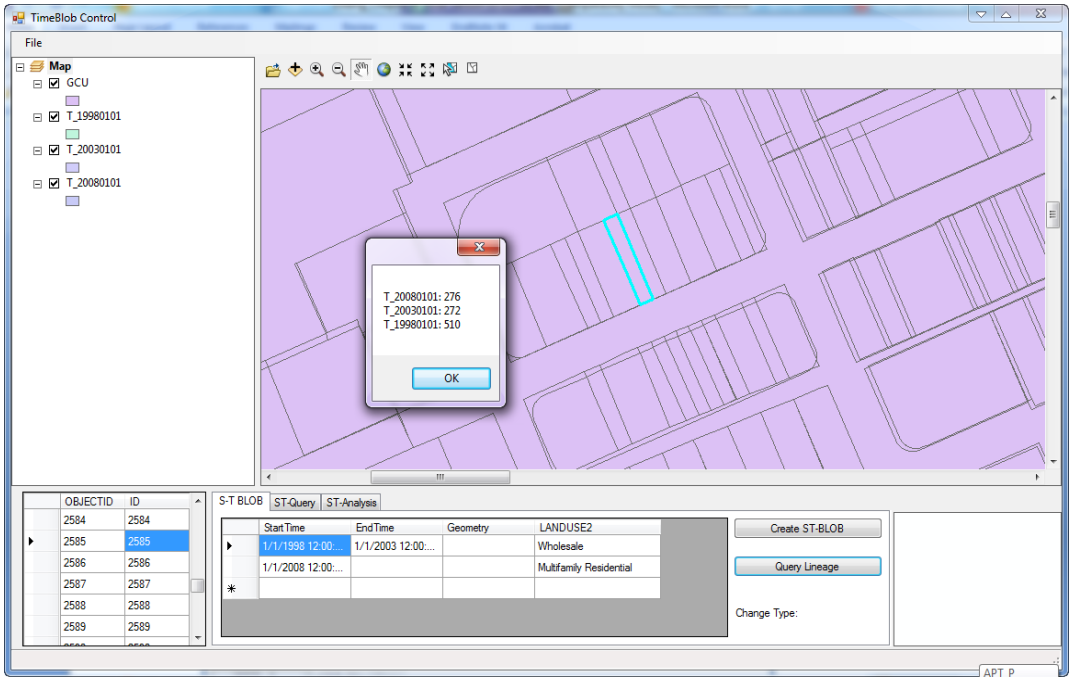


Figure 4.18 An example of the lineage of a greatest common unit

In another example of the greatest common unit which has OBJECTID 2592, this feature has one Space-Time Interval, which means that it exists from 1998 to 2008 as

multifamily residential (Figure 4.19). It belonged to the feature with ObjectID 471 in 1998, the feature with ObjectID 271 in 2003, and the feature with ObjectID 275 in 2008.

ESRI parcel model (see Section 2.2.10) also has the ability to query the lineage of the historical object by linking the records between the feature class that represents the current state and the feature class that stores the retired objects. In_Date and Out_Date of objects are used to create this linkage. However, when an object changes actively through time, users would have to switch back and forth etween current feature class and historical feature class to track changes of an object. By taking advantage of object-oriented design, this study could access the change history at the greatest common unit level very efficiently.

4.4 Functions of Spatial-temporal Query

The proposed space-time GIS data model organizes the complicated and redundant spatio-temporal data into a single compact Space-Time Feature Class. On the other hand, it allows comprehensive spatio-temporal queries to be performed in a much more efficient way than the snapshot model by accessing and analyzing the Space-Time Elements stored within the Space-Time BLOB. The object-oriented event-based space-time GIS data model could answer certain types of spatio-temporal questions very quickly. In the following section, six general types of spatio-temporal queries are implemented in this prototype system: (1) What -> When + Where; (2) When -> What + Where; (3) Where -> What + When; (4) What + When -> Where; (5) What + Where -> When; (6) Where + When -> What. These spatio-temporal query types cover the most common query scenarios. This section discusses how the prototype system can handle these different types of spatio-temporal queries, using the Chinese historical province example. Among these six types of spatio-temporal questions, the first three require only one parameter and show a list of answers in the other two parameters that meet the requirement. The last three need two parameters and answer the third one.

4.4.1 Interface design of Spatio-temporal query

Since there may be many spatio-temporal query questions with reference to users' specific needs, a general interface design is crucial, one that is capable of handling most spatio-temporal questions (Figure 4.20). Since the prototype system is designed to show people what it can accomplish, it is does not deal with each spatio-temporal query. However, this design should be flexible enough to be extended in order to answer additional questions in the future.

Within the spatio-temporal query tab are many controls organized in three panels: what, when, and where. The controls within these panels are grayed out before users select a query type from the dropdown list in the left.

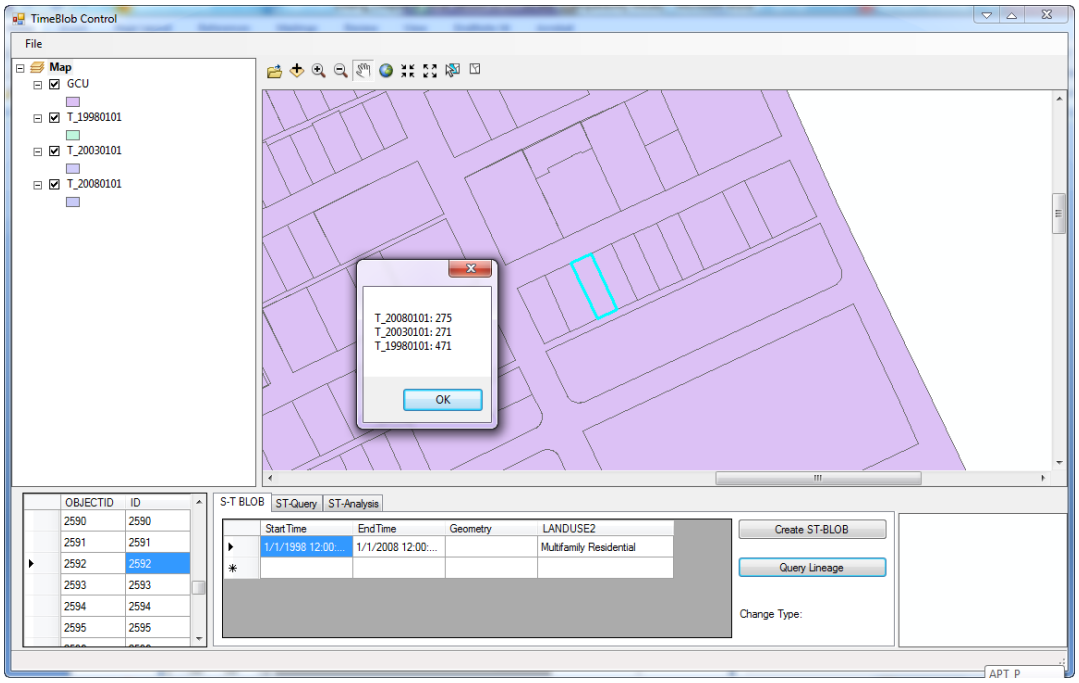


Figure 4.19 Another example of the lineage of a greatest common unit

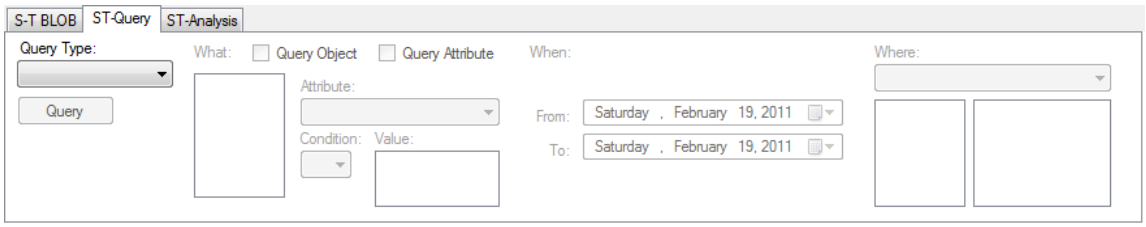


Figure 4.20 Interface design of spatio-temporal query

Corresponding panels, including every control within the panel, would be activated after users select a query type under the drop down list (Figure 4.21). Users could then specify the parameters and perform the query.

4.4.2 What -> When + Where

In this case, users specify one or more objects and query their spatial locations and geometries, as well as temporal information. This "what" can be further categorized as "object-based" and "attribute-based." For the former, users specify the ID of the Space-Time Feature. Spatio-temporal questions, such as "show the spatial location and geometry change of Beijing" belong to this type. For the latter, users specify the objects that meet specific standards, such as "show the spatial location and geometry of historical provinces with populations of more than 10,000,000."

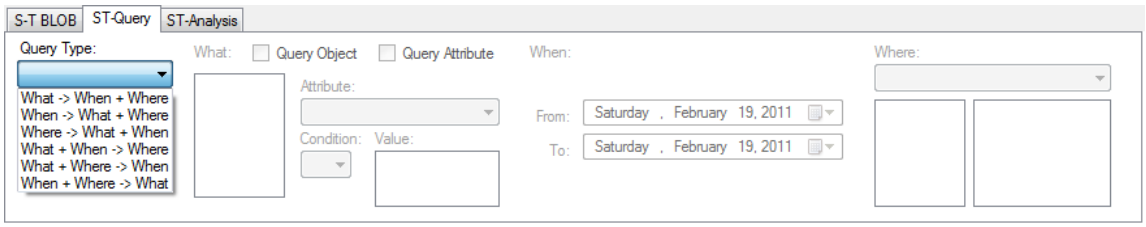


Figure 4.21 Query type selection

The operation and outcome of What -> When + Where begins with selecting What -> When + Where from the query type dropdown list, activating the "What" panel (Figure 4.22). Checking the "Query Object" checkbox populates an object list with entity IDs. Selecting Sichuan and clicking Query provides results.

Three feature classes are generated to represent the "when" and "where" information of Sichuan. The former is represented by the layer name, while the latter is represented by the features in the map. Users could turn the layer on and off to see the differences. The event-based data model enables users to determine when the selected object changes. In this example, Sichuan exists from 1953 to 2000: in its Space-Time BLOBs, Sichuan's first Space-Time Element is a Space-Time Point in 1953; in 1964, its administrative boundary changed, which is why it has a new Space-Time Element from 1964; the Space-Time Interval from 1964 to 1990 indicates that Sichuan Province remained consistent during this period. In the historical map of 2000, Sichuan Province changed its shape again, indicated by the blue area (Figure 4.23). In fact, this part became Chongqing city in 1997, which is under control of the Chinese central government. However, although the query result indicates that change occurred between 1990 and 2000, it is not clear in which exact year that this change happened. This is the major shortcoming of the snapshot model that historical data use. However, after creating Space-Time BLOB, it is possible to record the exact change time for each object when it changes by adding a Space-Time Element.

Users also could choose to specify "what" by entering the attribute requirement, checking the "Query Attribute" checkbox, selecting the Space-Time Features whose PROVGB (the numerical ID of the province) are less than 13, and querying "where" and "when" of these features (Figure 4.24). The results window displays two Space-Time Features—Tianjin and Beijing—that meet the requirement (Figure 4.25). Tianjin has two Space-Time Elements: one is a Space-Time Point in 1953, while the other is a Space-Time Interval from 1982 to 2000. If there is some question as to where Tianjin was in the map of 1964, Tianjin became a city of Hebei Province in 1958 and was designated as a municipality directly under the control of the Chinese central government in 1966, explaining Tianjin's absence in the map of 1964 and its existence in the map of 1982. The shape of Tianjin is also enlarged in the map of 1982. Thus, people cannot see Tianjin's Geometry in 1953 unless turning off the first layer. Beijing also has two Space-Time Elements, each of which represents Beijing's "where" and "when" information.

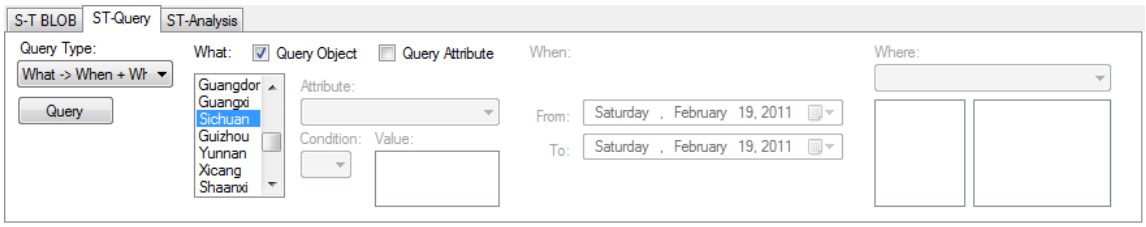


Figure 4.22 The settings of spatio-temporal query “What -> When + Where”

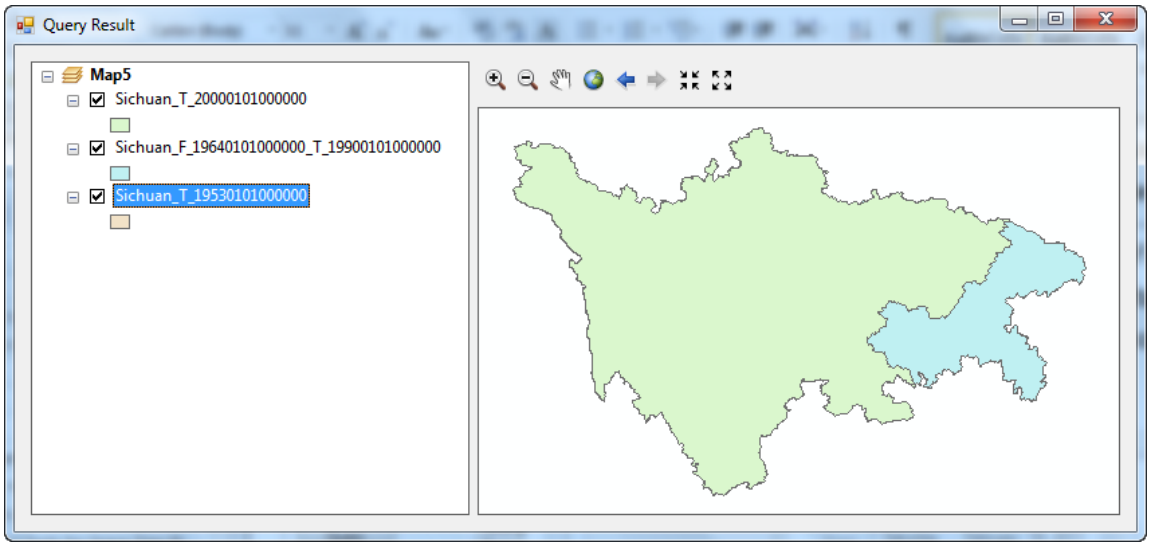


Figure 4.23 The result of spatio-temporal query “What -> When + Where”

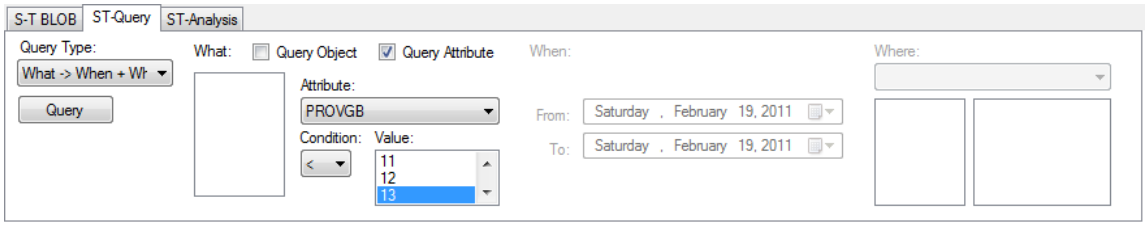


Figure 4.24 The settings of spatio-temporal query “What -> When + Where”

By using the suggested space-time GIS model, the speed of the query is accelerated since the specified object and its spatio-temporal information is accessed quickly. How each object changes and when changes happen is stored internally in its Space-Time BLOB. As a result, looping all the historical snapshots becomes unnecessary.

4.4.3 When -> What + Where

In this case, users specify the time period in which they are interested. A time period with identical start and end times represents a time point. The prototype system is able to list all the features that exist during this period, as well as their spatial locations and

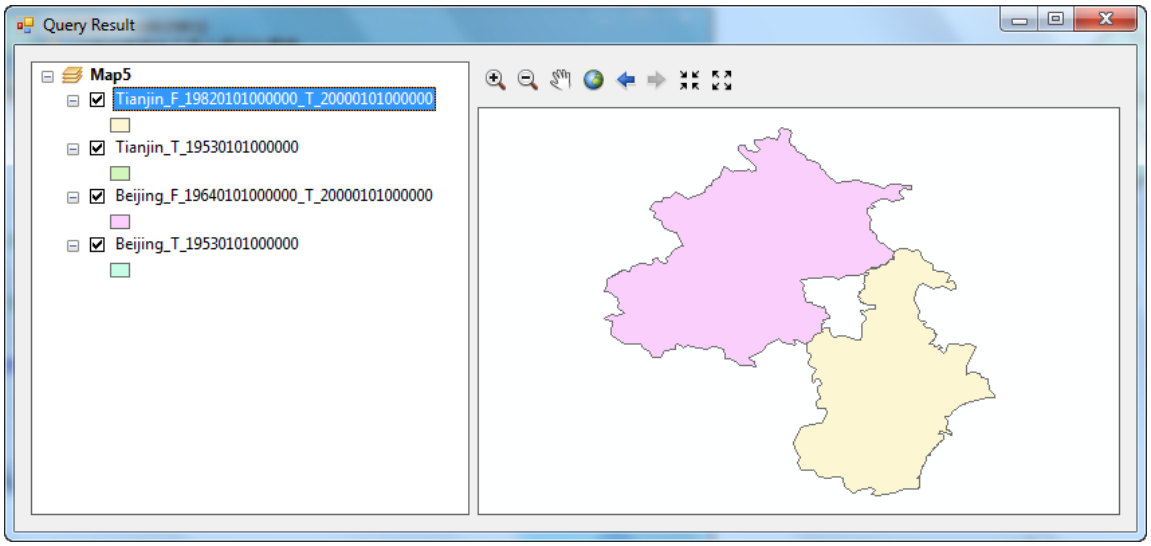


Figure 4.25 The result of spatio-temporal query "What -> When + Where"

geometries. Spatial location refers to the position in space, while geometry refers to the boundary.

To apply this type of spatio-temporal query, select When -> What + Where under the query type dropdown list to activate the "When" panel. The user would then specify a time period as the parameter and click query. This example queries all the features and their locations that existed during February 19, 1980 and February 19, 2011 (Figure 4.26).

The results show not only the objects that existed in the specified temporal range, but also from the layer information in the TOC how these objects changed (Figure 4.27). It is not uncommon that several provinces are listed more than once because the boundaries changed between February 19, 1980 and February 19, 2011. As a result, they have more than one Space-Time Element during this time period. For example, Macau changes twice during the specified time period so it has three Space-Time Points. It does not have two consistent states between two consecutive snapshots, which is why Macau does not have a Space-Time Interval. Xizang (Tibet) exists without any spatio-temporal changes and thus appears as a Space-Time Interval in the result.

The query "when -> what + where" is feasible using snapshot model; however, the answer would be a series of snapshot layers during the specified period. It is difficult to understand how each object evolves.

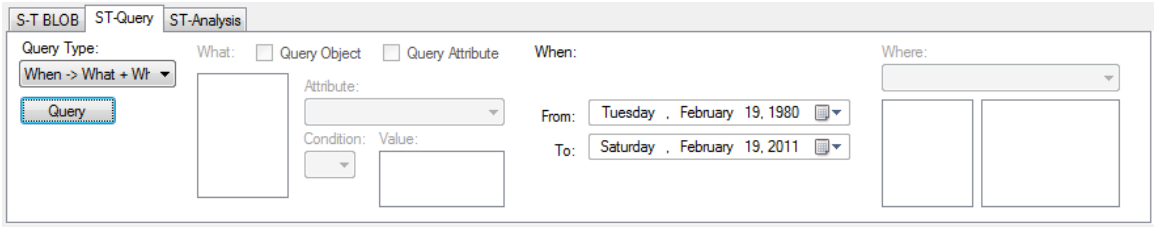


Figure 4.26 The settings of spatio-temporal query “When -> What + Where”

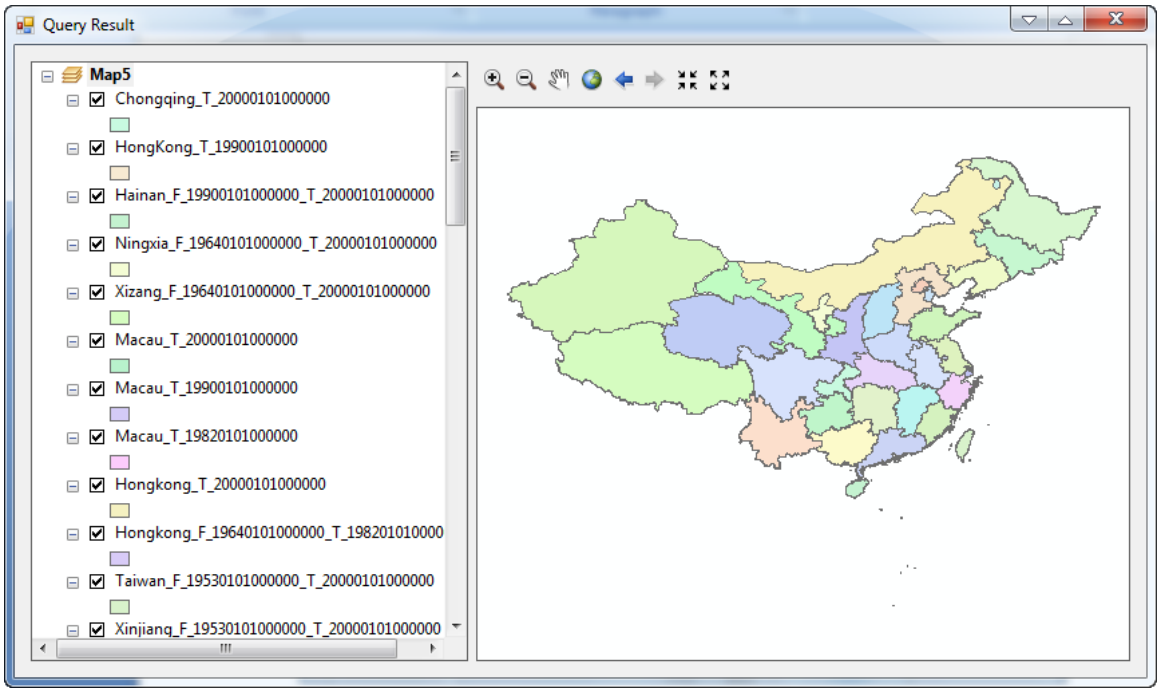


Figure 4.27 The result of spatio-temporal query “When -> What + Where”

4.4.4 Where -> What + When

In this case, users specify a location to see all objects that once existed in the location and their corresponding temporal information. Two options are available to capture “where”: users could drag a region on the map, or the region can be specified by its spatial relationship with other objects, such as “overlap with Beijing in 1980.”

Selecting Where -> What + Where in the query type dropdown list enables the controls in the “Where” panel. Users can then specify the spatial relationship the target objects have with another object, for instance, what features overlapped when with Beijing’s boundary in 1953 (Figure 4.28).

The result is straightforward: Beijing’s boundary from 1964 to 2000 overlapped itself in 1953 (Figure 4.29). Hebei Province also overlaps it from 1964 to 2000, but with the

difference that Hebei changed during this period. Consequently, it has two Space-Time Elements that meet the requirement.

4.4.5 What + When -> Where

The following three sections introduce other three types of spatio-temporal queries. For these three queries, two parameters are needed as inputs so they can be considered to be more stringent versions of the first three types of spatio-temporal queries. It should be mentioned again that the results not only display the objects themselves, but also how they change over time. On the other hand, the proposed Space-Time GIS data model is superior to snapshot model in that the snapshot model requires that objects from all snapshots be selected more than one time (e.g., one for "what" and one for "when"), and overlay them to get the final answer. However, "what," "when," and "where" all store in the Space-Time BLOB and, because selecting objects by analyzing the Space-Time BLOBs is sufficient, no overlay is needed.

For What + When -> Where, users are required to specify what objects they are interested in, as well as the temporal duration. After selecting the query type as What + When -> Where, both What and When panels are activated when, for instance, investigating Sichuan Province's location from February 19, 1980 to February 19, 2011 (Figure 4.30).

Sichuan's locations during 1980 and 2000 are clearly shown in the resulting map (Figure 4.31). Sichuan object has two Space-Time elements: one is a Space-Time Interval from 1964 to 1990, while the other is a Space-Time Point in 2000. We can also identify the area that became Chongqing municipality in the late 1990s.

4.4.6 What + Where -> When

In this case, users set a standard for selecting the objects in a certain location, and then the prototype system finds the corresponding objects and shows the users what time these objects exist, which can be read from the TOC. A typical example would be "when did Hebei Province overlap Beijing's geometry in 1953?" In this question, Hebei Province is

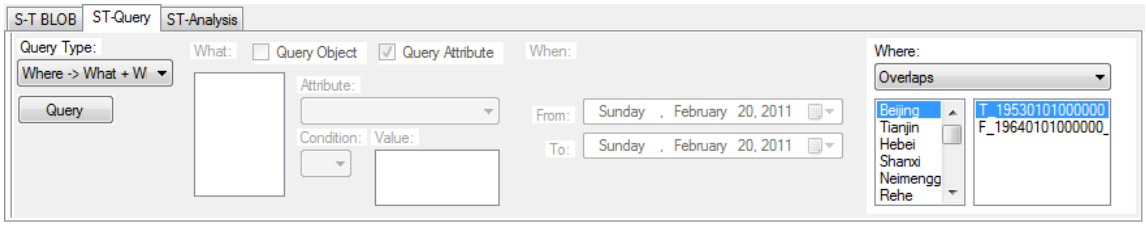


Figure 4.28 The settings of spatio-temporal query "Where -> What + Where"

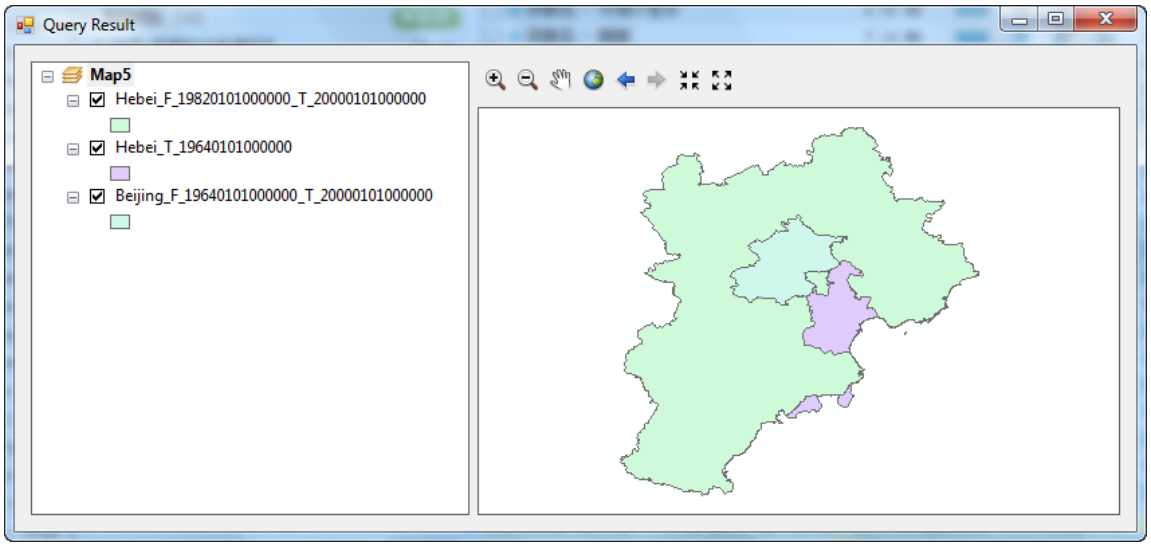


Figure 4.29 The result of spatio-temporal query “Where -> What + When”

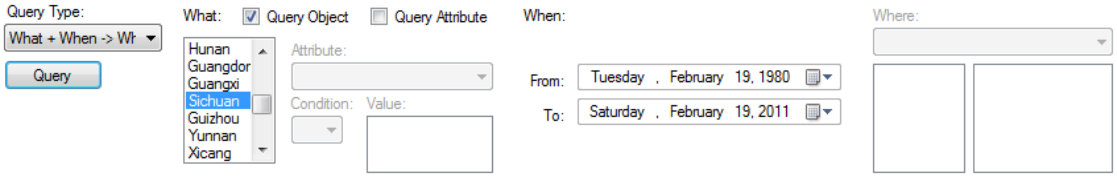


Figure 4.30 The settings of spatio-temporal query “What + When -> Where”

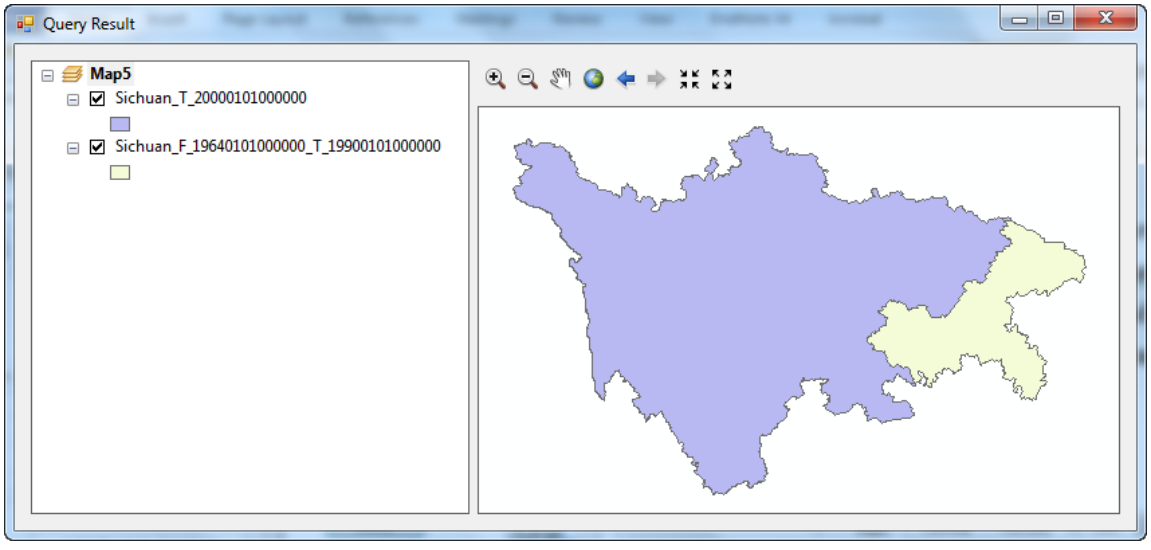


Figure 4.31 The result of spatio-temporal query “What + When -> Where”

the specified “what” and overlapping Beijing in 1953 is the specified “where” (Figure 4.32).

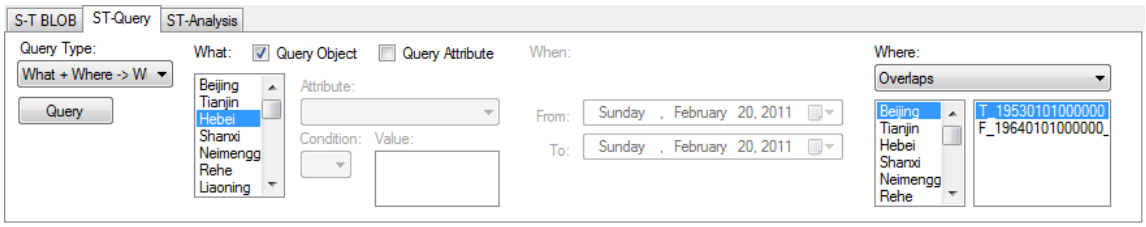


Figure 4.32 The settings of spatio-temporal query "What + Where -> When"

The result shows that Hebei Province's 1964 boundary overlapped Beijing's boundary in 1953 and overlapped itself from 1982 to 2000 (Figure 4.33). The reason for two layers in the result is that Hebei Province changed from 1964 to 1982, so two Space-Time Elements record the spatio-temporal information.

4.4.7 Where + When -> What

In Chinese history there once existed a province named Rehe. Rehe Province was split into several parts and merged with adjacent provinces in 1955. In order to answer the question, "What provinces from 1953-1964 overlapped with Rehe in 1953?" The "Where + When -> What" query is applied. Selecting this type from the dropdown list in the ST-Query tab enables the "When" and "Where" panels and sets the parameters (Figure 4.34).

The result shows that four provinces during that period overlapped with Rehe in 1953 (Figure 4.35). The question may arise as to why all of them start from 1964, but not 1955 when Rehe Province was revoked. The reason is that the snapshot in 1964 is the closest one to 1955 available. This again shows the weakness of the snapshot model. However, after storing the historical data in the Geodatabase using the proposed space-time GIS design, it is possible to record the new state for a Space-Time Feature when it varies so as to keep track of changes. Each object will have a Space-Time BLOB that tracks the spatio-temporal information of this object. At this point on, once a change happens on an object, this object's Space-Time BLOB can be edited by creating a new Space-Time Element that is able to precisely record the time of this change.

This prototype system described in this chapter is capable of creating Space-Time BLOBs for historical snapshot with/without entity identity, as well as general spatio-temporal queries. This object-oriented event-based design tracks spatio-temporal changes while minimizing the data redundancy. Historical snapshot could be handled very well by this prototype. In addition, the spatio-temporal query result gives users an idea about how each object changes over time.

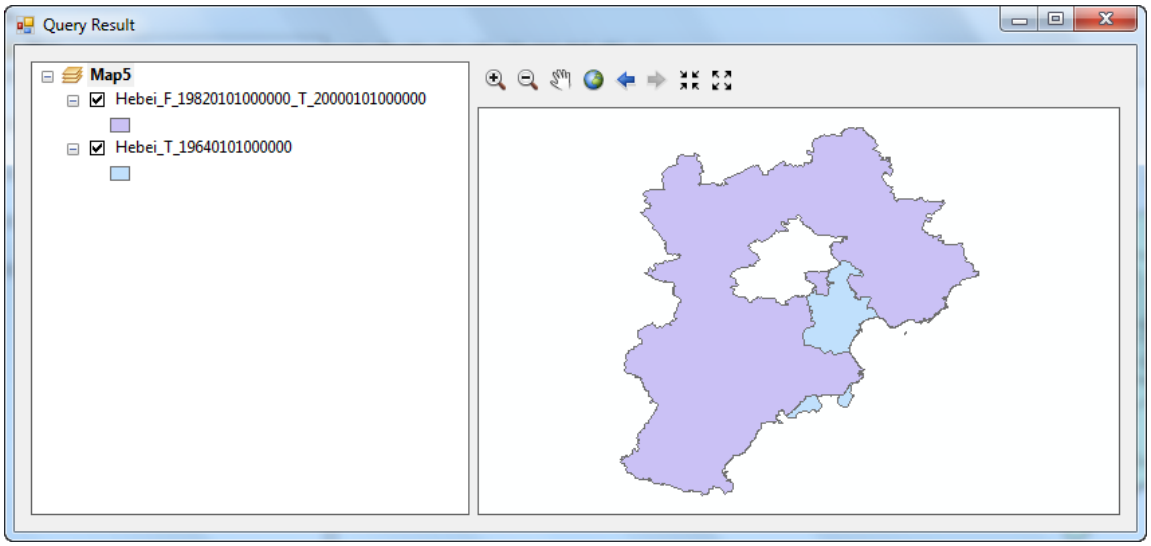


Figure 4.33 The result of spatio-temporal query “What + Where -> When”

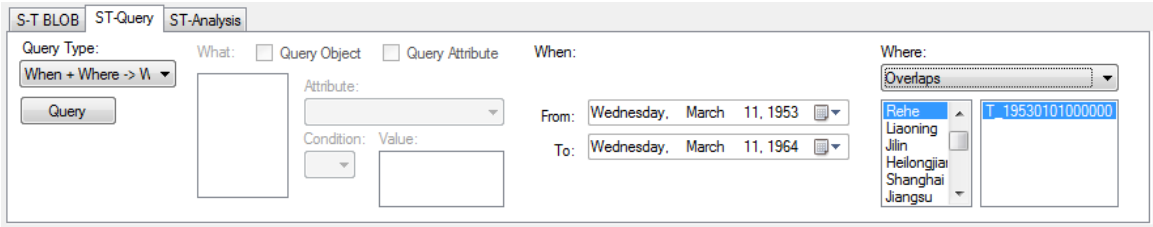


Figure 4.34 The settings of spatio-temporal query “Where + When -> What”

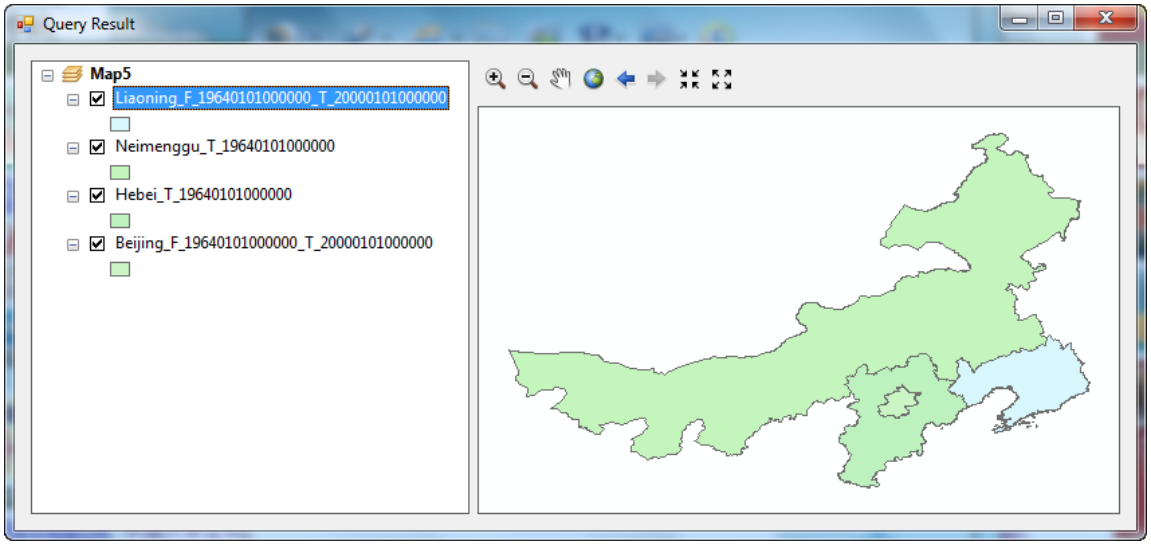


Figure 4.35 The result of spatio-temporal query “Where + When -> What”

CHAPTER 5 CONCLUSIONS

This chapter summarizes the achievements and contributions of this study. The researchers found many potential new topics that deserve further investigation. The accomplishment of this study is the starting point of a more unified framework. This chapter identifies several research topics that are likely to be addressed in the near future.

5.1 Summary of the Study

People in different areas are interested in the information related to temporal domain. Compared with previous decades, people now have more and more spatio-temporal data, which enable researchers to dig out hidden information accumulated over time to unveil the changes, patterns, and processes. Due to its significant data redundancy and limited spatio-temporal query and analysis capability, traditional snapshot model could not meet the requirements of handling increasing spatio-temporal data.

Emphasizing how to record each object's state over time, the proposed object-oriented space-time GIS data model abandons the timeline-oriented approach of snapshot models. It is flexible down to the object level since each object could have its own temporal arrangement with corresponding geometry and attributes. The Space-Time Interval and Space-Time Point, which are collectively labeled Space-Time Element, can be applied together to record each state of the object. The Space-Time GIS data model is also event-based. A new Space-Time Element would be generated and pushed into the Space-Time BLOB when triggered by a specific event. The design allows the storage of dynamic location, dynamic shape, and dynamic attributes; "dynamic" means these properties change over time instead of remaining static with constant values.

Researchers could apply the proposed Space-Time GIS data model from the beginning of data collection in order to organize the resulting spatio-temporal data. However, more often than not, we already have historical datasets that may have been accumulated over the course of a long time. A workflow is designed to cycle through each object in each historical snapshot and to create Space-Time BLOBs for the historical data. Two types of historical data were considered and addressed during this study: historical snapshot with entity identity and historical snapshot without entity identity, and different data processing procedures were invented for each.

Creating Space-Time BLOBs to organizing the historical dataset is not enough since data are continuously collected. The Space-Time BLOB should have the capability to incorporate the new data. Therefore, in addition to the mechanisms for creating Space-Time BLOBs from historical snapshots, this study also proposes a workflow to maintain the Space-Time BLOBs in the Geodatabase. *Snapshot model* could be replaced for object with entity identity data since data maintenance would be performed at the object level; for object without entity identity data we would still rely on snapshots, such as satellite

images, to update the data. However, the proposed workflow can handle the latest snapshot and re-create the Space-Time BLOBs at the greatest common unit level.

One practical consideration is how to store the Space-Time BLOBs so that they can be used for subsequent query and analysis purposes. This study addressed this problem by serializing the Space-Time BLOB in the BLOB type field of the Space-Time Feature to which it belongs. Deserialization could resurrect the Space-Time BLOB back into the memory when needed.

Six types of spatio-temporal queries were designed and implemented using the proposed Space-Time GIS design: (1) what -> when + where; (2) where -> what + when; (3) when -> what + where; (4) what + when -> where; (5) what + where -> when; and (6) where + when -> what. The performance of the spatio-temporal queries was satisfactory, especially when "what" is a query parameter that takes full advantages of the object-oriented design.

5.2 Future Research Directions

This study serves as the starting point of a larger project to be addressed in my PhD program. Several research directions are identified and discussed in this section.

5.2.1 Incorporating other forms of time

This study introduced time interval working together with time point, a design that enables a more flexible way to record each object. Actually, there are other forms of time. Many objects' schedules are time cycles such as the daily schedule of a bus and the class schedule of a student. If everything operates according to a predetermined schedule, the object must be present in the specified place at a given time. By assigning a schedule to the object, it is possible to determine the location of the object. A schedule would be a series of Space-Time Elements stored in the Space-Time BLOB. Each Space-Time Element represents an activity in this schedule, including the temporal information (a time point or a time interval), predetermined spatial location, and attributes within this time. The next activity of the last one in this schedule is the first activity in this schedule. No major challenges are involved when incorporating a time cycle.

A time branch is useful in representing more than one temporal arrangement if we do not have an exact idea about what is going to happen in the future or what has happened before. For example, a student might have three choices in the evening: hanging out with friends, studying at the library, or participating in sports in the gym. Time branches could represent the different time arrangements. Therefore, spatio-temporal queries or analyses can provide the answers in terms of the configuration of each branch. The key research challenge is how to design the data structure to store the time branches so that the prototype system is able to detect them and treat them as parallel time lines. A new class, the Space-Time Branch Class, must be developed to accommodate this

information. It could be stored in the Space-Time BLOB in conjunction with Space-Time Point and Space-Time Interval. Also, since a time branch represents a temporal arrangement, each Space-Time Branch could contain a series of Space-Time Points and Space-Time Intervals. Although complicated, this design appears to be feasible.

5.2.2 System implementation for point, polyline and raster

This study uses the polygon type data and scenario for the Space-Time GIS design. Point, polyline and image data storage and processing are not implemented due to the limited time. However, the space-time GIS concept suggested in this study has the potential to be applied on point, polyline and image data. Point, polyline data are somewhat similar to polygon data. Each point or polyline feature could be assigned a Space-Time BLOB with its spatio-temporal information in a BLOB type field. Point or polyline data can be with or without entity identity. For instance, point with entity identity data involves traffic count stations used to collect traffic volume data by department of transportation. The criminal maps which represent the location of criminal behavior could be a good example of point data without entity identity. Each layer represents a time period within which the crimes happened; the positions where crimes happened are recorded as points in this layer.

Raster, another major data type in GIS, consists of a matrix of cells organized into rows and columns. The data structure of raster dataset is much different than vector type data (i.e., point, polyline, and polygon). The most typical raster data is satellite imagery. Depending on the sensors and satellites, the imagery might have different spatial, spectral, and temporal resolutions. For instance, the spatial resolution of the MSS (Multispectral Scanner System) installed in the Landsat is 68m x 83m, the temporal resolution is 18 days on Landsat 1-3, 16 days on Landsat 4-5, while the image size is 185km x 185km. The characteristics of raster dataset involve: (1) each cell is a regular square; (2) cells do not move at different times so it is not necessary to store spatial locations of raster cells; (3) each cell must have a value for each band at different times; (4) there might be more than one band in a raster layer. The fundamental idea is to create a Space-Time BLOB for each pixel. This Space-Time BLOB stores the spatio-temporal information of the pixel by pushing Space-Time Elements into the Space-Time BLOB. The method used is similar to the one used for vector data: first create a Space-Time Point; convert it to a Space-Time Interval if no change happens; create a Space-Time Point again upon attribute changes; otherwise, extend the end time of the previous Space-Time Interval. All Space-Time BLOBs form a Space-Time Matrix which can be serialized and stored as a binary file in the hard drive.

One interesting research question arises about how an object can sometimes change its data type from one to another. For example, a hurricane can be seen as a point at the very beginning. After certain period of development, it could be treated as a polygon. Traditional GIS does not allow objects with different data types to be stored in a single layer. However, with the proposed Space-Time GIS data model, each state of an object appears as a Space-Time Element. The geometry is stored with the Space-Time Element,

and different Space-Time Elements are able to hold different types of data. Thus, the development of an object over time can be accurately recorded.

5.2.3 Incorporation with analytical time-geographic framework

Yu and Shaw (2008) proposed an *analytical time-geographic framework* based on Hägerstrand's (1970) *time geography* concept. They also extended it to a framework that can handle both physical and virtual activities. Under this framework, each person has from one to many polylines, each of which stands for an activity of this person. When undertaking the spatio-temporal query and temporal dynamic segmentation (i.e., locate individual activity events on spatio-temporal line features), all the records of all individuals must be looped through in order to obtain the ones that meet the requirement. If using the space-time GIS data model proposed by this study, each person would have one Space-Time BLOB in the Geodatabase. His or her locations, as well as the event information, are stored within this Space-Time BLOB, with each event represented by a Space-Time Element. Thus, the temporal dynamic segmentation could be performed with greater efficiency.

Moreover, in the current analytical time-geographic framework, the underlying base map is static. In fact, the objects in the base map could change over time. The spatio-temporal information of the base map could be stored by applying the model proposed by this study. Hence, the visualization of the space-time path and space-time prism could be improved.

5.2.4 Spatio-temporal analysis

The current research concentrates on the creation and maintenance of Space-Time BLOBs as well as the query function based upon the Space-Time BLOB, and in the future could move forward to spatio-temporal analysis. The key difference between spatio-temporal query and spatio-temporal analysis is that the former provides the user information from the data sources, while the latter uses the available data to generate new information. Typical analysis functions to be developed in the future may include: (a) identifying potential risk areas whose change patterns of environmental variables through a period correspond to a specific pattern; (b) analyzing the travel diaries of a large group of people and creating sub-groups within which they have similar behavior patterns and activity locations. Several interesting and useful scenarios would be identified and implemented to take advantage of the space-time GIS design for spatio-temporal analysis.

5.2.5 Processing power on very large dataset

Although this study has handled real-world data, and the performance of the prototype system is satisfactory, its ability to handle a very large dataset remains to be tested. It is expected that the performance would probably decline when the data volume explodes exponentially. This could be the case when encountering socioeconomic or ecosystem data that have been accumulated for a very long time. As a result, the system

optimization effort should be deemed necessary. The latest technologies from computer science can be referenced to assist the processing and analysis of the prototype system, such as parallel computing. Using parallel computing, a very large dataset can be divided into smaller ones, which are then processed concurrently while supported by multicore processors. This is an area that will most certainly be explored in the near future.

As discussed in this chapter, the Space-Time GIS data model proposed in this thesis study has already demonstrated its capacity to handle spatio-temporal data and spatio-temporal queries; it has great deal of potential to be extended. Also, although the current model design may contain flaws, its disadvantages and limitations would be identified when applied to real-world problems. Other researchers might also investigate its strengths and weaknesses. Thus, the research team led by Dr. Shih-Lung Shaw intends to keep making improvements in the model design, convinced that this space-time GIS design promises to be helpful to GIS researchers and experts in both academia and industry.

REFERENCES

ALLEN, J. F. 1984. Towards a general theory of action and time. *Artificial Intelligence,* 23(2), 123-154.
ARCTUR, D. and ZEILER, M., 2004. *Designing Geodatabases: Case Studies in GIS Data Modeling*. Redland, CA: ESRI Press.
ARMSTRONG, M., Temporality in spatial databases. ed. *Preceedings of GIS/LIS'88,* 1988 Bethesda, MD, 880-889.
AUFAURE-PORTIER, M.-A. and BONHOMME, C., 1999. A High Level Visual Language for Spatial Data Management. *In:* HUIJSMANS, D. and SMEULDERS, A. eds. *Visual Information and Information Systems.* Springer Berlin / Heidelberg, 653-653.
DARK, J. and BRAM, D., 2007. The modifiable areal unit problem (MAUP) in physical geography. *Progress in Physical Geography,* 31(5), 471-479.
ERWIG, M., *et al.* 1999. Spatio-temporal data types: An approach to modeling and querying moving objects in databases. *GeoInformatica,* 3(3), 269-296.
FRANK, A., 1998. Different Types of "Time" in GIS. *In:* EGENHOFER, M. J. and GOLLEDGE, R. G. eds. *Spatial and Temporal Reasoning in Geographic Information System.* New York: Oxford University Press, 40-62.
GOODCHILD, M. F. 1992. Geographical information science. *International Journal of Geographical Information Science,* 6(5), 407-423.
GOODCHILD, M. F., YUAN, M. and COVA, T. J. 2007. Towards a general theory of geographic representation in GIS. *International Journal of Geographical Information Science,* 21(3), 239 - 260.
GUO, D. S., *et al.* 2006. A visualization system for space-time and multivariate patterns (VIS-STAMP). *Ieee Transactions on Visualization and Computer Graphics,* 12(6), 1461-1474.
HÄGERSTRAND, T. 1970. What about people in regional science? *Papers in Regional Science,* 24(1), 6-21.
HODKINSON, I. and REYNOLDS, M., 2006. Temporal logic. *In:* BLACKBURN, P., BENTHEM, J. F. A. K. V. and WOLTER, F. eds. *Handbook of Modal Logic.* Amsterdam: Elsevier, 655-720.
HORNSBY, K. and EGENHOFER, M. J. 2002. Modeling moving objects over multiple granularities. *Annals of Mathematics and Artificial Intelligence,* 36(1-2), 177-194.
LANGRAN, G., 1992. *Time In Geographic Information Systems.* London: Taylor & Francis.
LAUBE, P., IMFELD, S. and WEIBEL, R. 2005. Discovering relative motion patterns in groups of moving point objects. *International Journal of Geographical Information Science,* 19(6), 639-668.
OTTO, R., 2010. *Mapping Time in ArcGIS 10* [online]. Available from: http://blogs.esri.com/Support/blogs/esritrainingmatters/archive/2010/09/08/mapping-time-in-arcgis-10.aspx [Accessed March 21st 2011].
PALMER, G. and BARKER, J., 2008. *Beginning C# 2008 Objects: From Concept to Code.* New York: Apress.
PEUQUET, D. J. 1994. It's about time - a conceptual framework for the representation of temporal dynamics in geographic information systems. *Annals of the Association of American Geographers,* 84(3), 441-461.

PEUQUET, D. J., 2002. *Representations of Space and Time.* New York: The Guilford Press.
PEUQUET, D. J. and DUAN, N. 1995. An event-based spatiotemporal data model (ESTDM) for temporal analysis of geographical data. *International Journal of Geographical Information Systems,* 9(1), 7-24.
PRASAD SISTLA, A., *et al.*, Modeling and querying moving objects. ed. *Data Engineering, 1997. Proceedings. 13th International Conference on*, 7-11 Apr 1997 1997, 422-432.
PRASAD SISTLA, A., *et al.*, 1998. Querying the uncertain position of moving objects. *In:* O, E., S, J. and S, S. eds. *Temporal database: research and practice.* New York: Springer.
PULTAR, E., *et al.* 2010. EDGIS: a dynamic GIS based on space time points. *International Journal of Geographical Information Science,* 24(3), 329 - 346.
RENOLEN, A., History graphs: Conceptual modelling of spatiotemporal data. ed. *GIS Frontiers in Business and Science*, 1996 Brno, Czech Rebublic.
ROBINSON, A., 2005. *Getting to know ESTAT: The Exploratory Spatio-Temporal Analysis Toolkit.*
SCHNEIDER, M., 2009. Moving objects in databases and GIS: state-of-the-art and open problems. *Research Trends in Geographic Information Science.* Springer Berlin Heidelberg, 169-187.
SHAW, S.-L. and XIN, X. 2003. Integrated land use and transportation interaction: a temporal GIS exploratory data analysis approach. *Journal of Transport Geography,* 11(2), 103-115.
SHAW, S.-L. and YU, H. 2009. A GIS-based time-geographic approach of studying individual activities and interactions in a hybrid physical-virtual space. *Journal of Transport Geography,* 17(2), 141-149.
SINTON, D. F., 1978. The inherent structure of information as a constraint to analysis: mapped thematic data as a case study. *In:* DUTTON, G. ed. *Harvard Papers on Geographic Information Systems.* Reading, MA: Addison-Wesley, 1-17.
SNODGRASS, R. T., 1995. *The TSQL2 Temporal Query Language.* Dordrecht: Kluwer Academic Publishers.
WANG, D. and CHENG, T. 2001. A spatio-temporal data model for activity-based transport demand modelling. *International Journal of Geographical Information Science,* 15(6), 561-585.
WOLFSON, O., *et al.* 1998. Moving objects databases: issues and solutions. *Scientific and Statistical Database Management, 1998. Proceedings. Tenth International Conference on*, 111-122.
WORBOYS, M. F., A model for spatio-temporal information. ed. *Proceedings: The 5th International Symposium on Spatial Data Handling,* 1992 San Jose, CA, 602-611.
WORBOYS, M. F., 1995. *GIS: A Computing Perspective.* Bristol, PA: Taylor & Francis.
WORBOYS, M. F., 1998. Modeling changes and events in dynamic spatial systems with reference to socio-economic units. *ESF GISDATA Conference onModelling Change in Socio-Economic Units.* Napthlion, Greece: Taylor & Francis.
YOUNG, M. and ZIMAN, J. 1971. Cycles in social behaviour. *Nature,* 229, 91-95.

YU, H. and SHAW, S.-L. 2008. Exploring potential human activities in physical and virtual spaces: a spatio-temporal GIS approach. *International Journal of Geographical Information Science,* 22(4), 409 - 430.
YUAN, M. 1999. Use of a three-domain representation to enhance GIS support for complex spatiotemporal queries. *Transactions in GIS,* 3(2), 137-159.
ZEILER, M., 2000. *Modeling Our World: The ESRI Guide to Geodatabase Design.* Redland, CA: ESRI Press.

VITA

Ziliang Zhao was born and grew up in Fuzhou City, China. He received a Bachelor of Science degree in 2009 and entered the University of Tennessee, Knoxville in the same year to pursue a Master's Degree in the Department of Geography. Since then, he has been studying space-time GIS and GIS for transportation under the guidance of his advisor, Dr. Shih-Lung Shaw. He will continue his work and pursue a PhD degree in the same department in August 2011.