CHAPTER FOUR

The spatio-temporal data model

> The term representation means an arrangement or organization of data defined within an explicit set of primitive elements [entities], attributes [properties], and relationships. Such an arrangement serves to preserve the information inherent in the data for subsequent use in problem-solving or analytical evaluation. A representation can be purely conceptual, but any representation must be expressed in formalized mathematical or programming terms for computer implementation. A formalized representation intended for computer implementation is known as a *data model.*
>
> D.Peuquet

Developing our spatio-temporal data model (STDM) based on Time Geography and object orientation has certain implications that must be taken into consideration independently of the knowledge domain we are hoping to represent:

- The spatio-temporal data model emphasises the interaction between events and states rather than a linear connection among them over a space-time path.
- The space-time path offers an appropriate semantic abstraction for handling only valid time in a spatio-temporal data model.
- The space-time path requires a different version management approach for incorporating change into a database. Most of the approaches extend the relational model by creating new versions of tables, tuples or attributes to reflect changes occurring over time. A new version management approach is necessary to deal with changes that are represented by the synchronisation of events and states.
- Object-oriented analysis and design is required for the development and implementation of the spatio-temporal data model within a GIS.

The strengths of the STDM and its implementation lie in the well-defined concepts and representations developed in Time Geography and object-oriented approaches. The spatio-temporal data model offers much practical guidance, and implies the

feasibility of applying object-oriented methods to the problem of handling space and time in GIS. This chapter gives a detailed description of the STDM used in the rest of this book.

4.1 DEFINING THE REASONING TASK

Spatio-temporal data modelling is about organising abstract concepts of a knowledge domain into formal descriptions. A spatio-temporal data model should provide constructs for representing a knowledge domain in terms of entity/event abstractions, relationship abstractions, behaviour specifications and interaction descriptions. The STDM is primarily constructed from the viewpoint of the fused space and time dimensions rather than the dimensions themselves and their orthogonality. The space-time path is treated as the core of the manifestation of space and time within this model. Therefore, each entity (real-world phenomenon, feature or object) from a knowledge domain has its own space-time path that represents its lifespan. However, the possibility of describing more than one space-time path for a singular entity is conceivable and can demonstrate the complexity that one can achieve in the STDM configuration. Shoham and Goyal (1988) distinguish four different reasoning tasks for developing space-time paths within an STDM:

- *Prediction* Given a description of the world over some period of time, and the set of rules governing change, predict the world at some future time.
- *Explanation* Given a description of the world over some period of time and the rules governing the change, produce a description of the world at some earlier time.
- *Learning new rules* Given a description of the world at different times, produce the rules governing change which account for the observed regularities in the world.
- *Planning* Given a description of some desired state of the world over some period of time and given the rules governing change, produce a sequence of actions that would result in a world fitting that description.

The first step towards designing space-time paths within an STDM is to uncover which task is deemed to be achieved. These tasks represent the overall goals associated with space-time paths of the STDM, rather than a delineation of the appropriate elements of a particular space-time path. The key issue here is to understand the spatial, temporal and semantic aspects of the data (or the knowledge domain) in relation to space-time paths. How data are expected to be explored in a search for spatial, temporal and spatio-temporal patterns is vital when delineating a conceptual configuration for designing space-time paths within the STDM; this chapter uses the second task—explanation.

Table 4.1 Main abstractions defined for a space-time path.

Elements	Main abstraction
Space-time path	Fusion of space and time dimensions
States	Changes produced by the connectivity and continuity of a sequence of events
Events	Human activity, the cause for one state change to another, or part of a process
Episodes	Length of time, duration
Evidence	Maps, surveys, satellite images

4.2 THE SPACE-TIME CONFIGURATION

A space-time path can have a longitudinal or branching configuration in which states and events, episodes and evidence are consecutively and chronologically connected through this path. Table 4.1 summarises the main abstractions involved in each of these elements of a space-time path. The origin of a space-time path can be associated with an event element or a state element. For instance, the *creation* of a space-time path can occur by a human action or by the explicit choice of a starting point of interest in an entity state at any particular time. In the STDM both events and states have been modelled as classes of objects. However, they play different roles within the STDM. Events are instances of classes that describe what and when something happened, is happening, or will happen during the lifespans of entities. On the other hand, states are instances of classes that describe what has changed, is changing, or will be changed during the lifespans of entities. The main advantage of this modelling decision is that events can be defined, exist within the database, and interact with the classes of states without depending on the changes in the states themselves.

The longitudinal or branching configuration of a space-time path encapsulates the existence of an entity over space and time. The *existence* will be a sequence of states and events in which change can occur over its states. A change can result from the effects of human activity due to an alteration, modification or transformation of an entity state over time. A space-time path will be an arrangement of events and states that have to be effectively connected to each other to represent the evolution of an entity lifespan. Exploring the temporality and dynamic interactions of classes of events and states may provide an in-depth understanding of our perceptions of the knowledge domain.

The observer's view is a very important factor in determining the spatio-temporal representation. The observer's position on the space-time path will determine their present, hence their past and future. As the observer moves along the space-time path, they experience changes in their perception and understanding of what is past, present and future. The actual notion of present, past and future in the STDM relies on the

location of the observer on the space-time path, not the space-time path itself. The historical view of the observer is reduced to the perception of nearness of states and events on the space-time path of a lifespan of an entity.

The demise circumstance characterises the closure of a space-time path. It can occur on the space-time path at any time during the lifespan of an entity. The effectiveness of the STDM in handling the creation, evolution and demise circumstances that can occur on a space-time path is fostered by the instantiation of object classes. An object of a class exists in time and space, whereas a class represents the essence of this object. An object of a class can represent either an event or a state in the STDM.

In the STDM, classes can be categorised as being generic, versioned and unversioned. Each property defined within a class is further classified as being an invariant, version significant or non-version significant attribute. Note how this classification is not deemed to embrace the infinite range of possible changes that can occur over entities of a knowledge domain. The proposed classification is based on changes that can be associated with three update procedures defined in the STDM: (a) creation of a new object, (b) creation of a new object from an existing object, and (c) relocation of an existing object (Table 5.3 gives the complete list of update procedures).

4.3 DATA MODEL CHANGES

The main finding in this investigation is that, although they are primarily concerned with updates in the schema of the spatio-temporal data model, the data model changes also affect the configurations of space-time paths. Data model changes affect the position, orientation and scale of a space-time path in the STDM. This influence has been observed during practical implementation of the STDM in Smallworld GIS. A general overview has evolved from these empirical observations as an attempt to demonstrate how data model changes can affect a space-time path.

In summary the following components of the Banerjee *et al.* classification (see Chapter 3, Section 3, for a detailed description) appear to be important in the present context:

1 Changes to the component of a class
   (a) Changes to properties: they affect the position of a space-time path
   (b) Changes to methods: they affect the orientation of a space-time path
2 Changes to relationships of classes: they affect the orientation of a space-time path
3 Changes to classes themselves: they affect the position and the orientation of a space-time path

A more detailed investigation is usually made into data model changes and how they affect the position, orientation, and scale properties of a space-time path, but it is beyond the scope of this book.

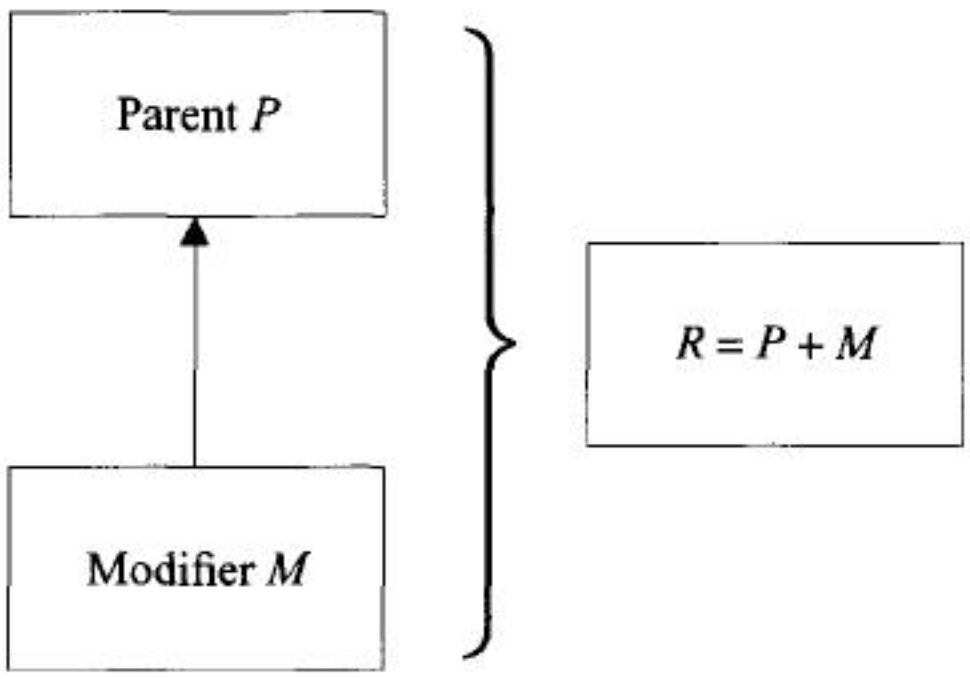

Figure 4.1 Independent incremental modification by inheritance

4.4 CONNECTIVITY ASPECTS

The STDM provides a distinction in version configuration between the system level and the application level. At the system level, version management is supported by object versions that reflect the history of modifications over objects as perceived by the system. At the application level, version management supports the representation of valid time or the sequential development of information as defined by the user or the application.

Considering that states and events from the knowledge domain will be classes of objects within the STDM, the design of an incremental mechanism relies on defining the kind of relationship that should exist among these classes in order to reproduce a space-time path. In fact, this involves an instance of one class being connected with an instance of another class in such a way that it reproduces the history of a particular entity on a space-time path. This has been achieved by using the *inheritance* abstraction to develop an incremental mechanism for the space-time path. Changes happen at the instance level in such a way that an entity has its space-time path depicted by different instances of different classes of objects representing the events and states, which are connected through a mechanism of incremental modification.

Inheritance has been adopted as the incremental modification mechanism for designing space-time paths within the STDM due to its potential for subclassing the evolutionary processes of incremental change of the Time Geography approach. Two types of incremental modification mechanism can be described in the STDM, the independent incremental modification and the overlapping incremental modification.

An *independent incremental modification* by inheritance transforms a parent class P with a modifier M into a resultant class $R=P+M$ (Wegner and Zdonik, 1988). The composition operator+plays an asymmetric role in determining R from P and M (Figure 4.1), since

result R
    inherits P;
    modified by M.

The parent, modifier and result classes exhibit a structure with a finite number of properties:

$$P = (p_1, p_2, \ldots, p_p)$$

$$M = (m_1, m_2, \ldots, m_m)$$

$$R = (r_1, r_2, \ldots, r_r)$$

In the spatio-temporal data model, the independent incremental modification occurs over object classes belonging to the creation, existence and demise circumstances of a space-time path. It represents the connection state-event-state of a space-time path. In this case the parent class is an evolutionary state that is modified by an event (modifier class) into another evolutionary state (resultant class). The properties of the resultant class ($r_1, r_2, \ldots, r_r$) are independent of those properties of the parent class ($p_1, p_2, \ldots, p_p$) and the properties of the modifier class ($m_1, m_2, \ldots, m_m$). The independent incremental mechanism is responsible for the union of these properties in a way that R has $r+p+m$ properties.

Conversely, an *overlapping incremental modification* by inheritance transforms a parent class into a resultant class without having the presence of a modifier class. The properties of the resultant class are inherited from the properties of its parent class (Wegner and Zdonik, 1988). In STDM the overlapping incremental modification occurs over versioned and generic classes. In this case the resultant class is assigned to an instance of the class which inherits some of the properties of its parent class. New properties can be added to the resultant class whose names do not occur in its corresponding parent class.

In conclusion the space-time path is built by connecting each instance of a class with its corresponding instance of another class, according to the incremental modification mechanism involved. This allows us to represent the connection between events and states within the spatio-temporal data model by using the instantiation of object classes and the inheritance relationship between them. Having such a configuration for space-time paths in the spatio-temporal data model means there is a fundamental need for version management of all instances that can belong to a space-time path. This implies the analysis and design of a version management mechanism in order to manage change over time within the spatio-temporal data model.

4.5 VERSION MANAGEMENT

The version management mechanism designed for the STDM implies versioning at the instance level of versioned classes, as well as the classification of their attributes. Versioning at the instance level (object level) uses the available structure of classes in the STDM. In this case, version significant attributes have been implemented by attaching an update method to each one of them. This is achieved by defining trigger

update methods for each version significant attribute of these versioned classes. If a user tries to update a version significant attribute, a menu pops up on the screen informing the user to update the version significant attribute in a non-destructive manner. Each version is implemented as a record and methods can be invoked on this record in order to perform the update operations on version significant attributes (see Chapter 3, Section 3.5). The visualisation of versions is achieved by the use of pop-up windows which display a version at the instance level of a versioned class. This is illustrated in the next chapter.

Non-version significant attributes have been incorporated into the STDM in such a way that they are attributes whose values are updated in a destructive manner. Versions are not stored in the database. Invariant attributes have been incorporated into the STDM by attaching an update constraint to the instances of the classes. This is achieved by defining a trigger method for each invariant attribute (record) in a way that its value is not allowed to be updated. This method is triggered every time a user tries to update invariant attributes.

Ordering versions within a spatio-temporal database is also an important aspect in version management. Despite the development of storage devices, such as optical disks, offering new capabilities for storing large amounts of data, a significant waste of storage space will still be minimised by ordering versions compactly. The proper choice of a method for ordering temporal data relies on producing the best performance for a given application. Two general approaches have been developed for databases (Dadam, Lum and Werner, 1984). The first one is the absolute representation for ordering temporal data in which snapshot series are created for each version. The second is the relative approach in which versions are described in relation to a base state that can be located in the present or the past. In both approaches, *version identifiers* are adopted to address versions, and *delta versions* to store a version as a delta version instead of a complete version. Therefore, $Cv_x(n)$ is the version identifier n of the object x, with n_o being the oldest version and Cv standing for complete version. The delta version is defined as $\Delta_x(k, k')$ where Δ_x is the delta version between the versions number k and k' of an object x. Here are some relevant conclusions on the absolute representation:

- Based on complete versions.
- Versions $(x) = [Cv_x(n), Cv_x(n-1), \ldots, Cv_x(n_o)]$.
- Storage space requirement is prohibitive.

Relative representations are divided into backward and forward. The backward relative representation reconstructs older versions from newer versions:

(a) Backward-oriented
   - Versions $(x) = [Cv_x(n), \Delta_x(n, n-1), \Delta_x(n, n-2), \ldots, \Delta_x(n, n_o)]$.
   - It allows fast access to the current version. Unfortunately, in this versioning strategy, versions have to be recomputed completely whenever a new version is created. This results from the fact that deltas are related to the current version which changes whenever a new version is created.

(b) Backward-oriented accumulative

- Versions $(x) = [Cv_x(n), \Delta_x(n, n-1), \Delta_x(n-1, n-2), \ldots, \Delta_x(n_{o+1}, n_o)]$.
- All versions except the current one are expressed as delta to the successor-in-time version.
- Access time to versions will increase with their age.

The forward relative representation reconstructs newer versions from older versions:

(a) Forward-oriented

- Versions $(x) = [\Delta_x(n, n_o), \Delta_x(n-1, n_o), \ldots, \Delta_x(n_{o+1}, n_o), Cv_x(n_o)]$.
- Access time is the same for every version. Each version can be accessed in two steps because all deltas are related to the base version.

(b) Forward-oriented accumulative

- Versions $(x) = [\Delta_x(n, n-1), \Delta_x(n-1, n-2), \ldots, \Delta_x(n_{o+1}, n_o), Cv_x(n_o)]$.
- Access time to versions will decrease with their age.

Based on this evaluation by Dadam, Lum and Werner (1984), the backward-oriented accumulative strategy has been adopted in the STDM. The main advantage seen in this strategy is that for each version in the STDM, an accumulative delta is obtained due to a given update. In other words, the delta creation for obtaining the successor-in-time version in the space-time path can be activated by the occurrence of a change within the STDM. Therefore, a successor-in-time version can only be created if an update is triggered within the system. The visualisation of this strategy can be found in the next chapter, where the results are presented.

4.6 CONCLUSIONS

The abstractions developed in the STDM, such as space-time paths incorporating events, constraints and states, can be implemented into GIS. They can be used in modelling a variety of applications in environmental information systems, land information systems and other information systems. One example is the prediction of environmental change due to long-term large-scale climatic variations (Wachowicz and Broadgate, 1993). In this particular knowledge domain, prediction of environmental change requires an understanding of the principal mechanisms implicated in long-term large-scale climatic variation. Uncovering these mechanisms can only be achieved by analysing past environment states as well as recognising patterns of change through time. The STDM provides the semantics of events and states that can be used to describe the environmental changes. They can also be categorised as effect (an environmental change which can be detected by experience or observation of the environment) and cause (circumstances acting over a period of time which produce an environmental change). The STDM proposed in this book enables the time geographic framing of past environmental states (changes) by exploring the derivation of events from these states, and vice versa, due to their spatio-temporal interdependence.

Evaluating a spatio-temporal data model is a difficult and drawn-out process. The approach taken in this book has been to construct a prototype implementation as complete as possible within the time and resources available and to use it for evaluating the feasibility of the STDM, along with its strengths and weaknesses. This is discussed in more detail in the next chapter. A complete evaluation, in particular an implementation of a whole application system, would take much more time than was available.

The results of the implementation are presented in the next chapter and show that the prototype implementation preserves the principal features of the STDM. Care had to be taken when implementing these features, such as space-time paths and their associated incremental modification mechanism, to preserve the functionality of the spatio-temporal data model. The resulting implementation is therefore well placed to take full advantage of the object-oriented constructs (object identifier, inheritance, polymorphism).

The strengths of the STDM and its implementation lie in the well-defined 'object representation' that has been developed. This representation offers much practical guidance, and implies the feasibility of applying object-oriented methods to the problem of handling space and time in GIS. The Time Geography approach brings to light the link between components as events, states and constraints, and also much of the functionality of modelling space and time in the context of the knowledge domains in GIS.

The weaknesses lie in the lack of an appropriate access method for implementing space-time paths. New indexing techniques need to be developed in order to avoid problems in the implementation of these paths. This problem is not addressed here, but many new indexing techniques have been proposed in the literature (Schneider and Kriegel, 1992; Renolen, 1996).

Although this research concentrates on spatio-temporal data modelling, it is hoped the findings will contribute more widely to the next generation of geographic information systems, with their improved capabilities for handling spatio-temporal data. The STDM presented in this book emphasises many desirable characteristics for a temporal GIS:

- A continuous space-time path that supports modelling states and events in order to integrate space and time in a GIS.
- A sensible compromise between the flexibility offered by object-oriented methods and the drawbacks of implementing an object-oriented data model in a GIS.
- Scope for handling changes within a GIS.

But further research is necessary, especially in these three areas:

- To develop new indexing techniques for spatio-temporal data.
- To investigate new approaches for storing and manipulating topological relationships in a GIS, in order to avoid adding unnecessary complexity to the spatio-temporal data model.
- To develop temporal capabilities such as dynamic visualisation and spatio-temporal queries.