3 Changes to classes themselves
- Add a new class
- Drop an existing class
- Change the name of a class

Data model changes describe the evolution of object-oriented data models that involve updates to the schema. They are not related to the update procedures on the data available in the database. They can be considered as changes that need to be performed in order to rectify errors encountered in the schema of a data model. Otherwise, they can also be related to the changes which are necessary due to the development of our understanding of the knowledge domain.

Data model changes play an important role in a distributed GIS environment with a large number of users. A version management mechanism has to be provided so that a designer can test any data model change to the data model in their own version of the whole database. Some GIS products present a mechanism for handling data model changes, e.g. Smallworld GIS. The CASE tool[16] in Smallworld GIS has been specifically developed for handling data model changes. The version management implemented in the CASE tool has the ability to define and modify object classes: 'This uses version management techniques extensively to help functions like the ability to create alternative versions of the data model, apply these to alternative versions of a populated database for testing and development, and finally apply the changes to the master version of the database with a mechanism for propagating the schema changes down to other versions in the database' (Newell and Batty, 1993, p. 3.2.3).

3.4 TEMPORAL DATABASES

By the end of the 1980s, researchers had recognised the need for databases to support information that varied in time (Ackoff, 1981; Anderson, 1982; Ben-Zvi, 1982; Bonczek, Holsapple and Whinston, 1981; Clifford and Warren, 1983; Snodgrass, 1987). However, certain misconceptions had arisen concerning the terminology and definition of the features supported by temporal databases. The work of Snodgrass and Ahn (1985) was the first to present a new taxonomy of time that unified the concepts developed in the literature.

Transaction time has been proposed as a consensus term for previously defined physical time (Dadam, Lum and Werner, 1984; Lum *et al.,* 1984), registration time (Ben-Zvi, 1982), data valid from/to (Mueller and Steinbauer, 1983) and start/end time (Reed, 1978). Correspondingly, valid time has been proposed for event time (Copeland and Maier, 1984), effective time (Ben-Zvi, 1982), logical time (Dadam, Lum and Werner, 1984; Lum *et al.,* 1984), state (Clifford and Warren, 1983) and start/end time (Jones, Mason and Stamper, 1979; Jones and Mason, 1980).

The terms 'transaction time' and 'valid time' have also been proposed to differentiate the distinct types of database according to their ability to handle temporal information.

[16] CASE tool is an interactive, visually oriented tool for creating and managing the database schema using graphical notations.

Four types of database have been defined according to their ability to support these time concepts and the processing of temporal information (Snodgrass and Ahn, 1986):

- *Snapshot databases* provide a unique snapshot of a database state.
- *Rollback databases* provide a sequence of snapshot states of the database by offering support for transaction time (the time the data are stored in the database).
- *Historical databases* provide the knowledge about the past by supporting valid time (the time the data have changed in the real world).
- *Temporal databases* support both transaction time and valid time.

Using the taxonomy of time employed in the research on temporal database systems, object-oriented databases can be considered as rollback databases due to their ability to represent temporal information. Object-oriented databases support the history of the transaction results in snapshot states rather than the history of the real world as we perceive it. Most object-oriented databases store all past states, indexed by transaction time, of the snapshot database as it evolves. Object-oriented databases may in the near future advance into the realms of temporal databases.

POSTGRES embodies the first substantive proposal for implementing a rollback database using optical disks with support for transaction management and concurrency control mechanisms (Stonebraker, 1987). In providing these extended database functionalities, the possibility of implementing spatio-temporal constructs as extended database functionality for POSTGRES has also raised expectations in the GIS field. The GEO System (van Hoop and van Oosterom, 1992) is an example of a prototype design in which spatial constructs have been implemented by developing some extensions to the POSTGRES structure. Probably the best-known effort to implement a spatio-temporal data model using the POSTGRES database is the Sequoia 2000 Project, led by Michael Stonebraker of the University of California at Berkeley (Gardels, 1992). POSTGRES has been integrated with the GRASS[17] GIS to allow temporal manipulation of global change data.

The temporal database management system (DBMS) prototype developed by the IBM Heidelberg Scientific Centre, in the Advanced Information Management Project, was the first attempt to implement both valid time and transaction time with the support of temporal indexing (Dadam, Lum and Werner, 1984; Lum *et al.*, 1984). This attempt has been important in consolidating the concepts developed by temporal research in databases, and it has made people aware of the need for spatio-temporal indexing in the GIS field. For example, Langran (1988) pointed out spatio-temporal indexing as a temporal GIS design trade-off, in order to define how to control the volume of spatio-temporal data required in an application within a GIS. To achieve spatio-temporal indexing for a temporal GIS, Hazelton (Hazelton, Leahy and Williamson, 1990) discusses a natural extension of a quadtree structure; and most appropriate, he suggests, could be a multidimensional indexing structure.

[17] Geographic Resources Analysis Support System.

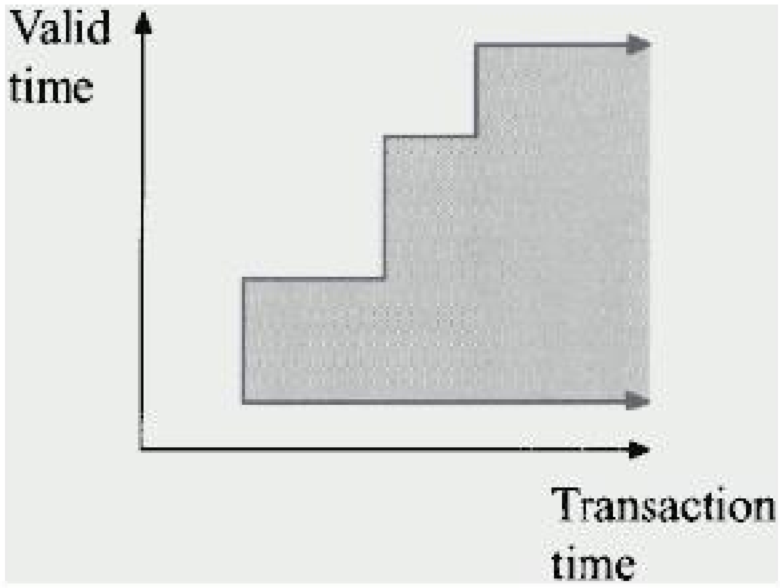

Figure 3.1 The bitemporal element

Snodgrass and Ahn (1985, 1986) have also demonstrated the existence of a true orthogonality between transaction time and valid time which allows each kind of time to be pursued independently: 'Time is a universal attribute in most information management applications and deserves special treatment as such' (Snodgrass and Ahn, 1986, p. 35). As a result, Snodgrass (1992) subsequently introduced the concept of a bitemporal element. This element is deemed to represent the valid time-transaction time space (Figure 3.1).

Several applications in GIS require information on the state of an object when it has been modified in the database to match its change in the real world. Worboys (1994) investigated the possibility of creating bitemporal elements for spatial chains (line objects) in object-oriented databases. The implementation used a discrete model for linear time. His findings show that the critical areas for research are the development of appropriate spatio-temporal data models, the indexing technique for this model, and related performance issues.

By looking for innovative directions in the temporal research domain, Snodgrass (1990) alludes to the outstanding potential of object-oriented databases which offer significant support for handling time, despite the lack of research work carried out in object-oriented databases about valid time. The important role of object orientation has also been identified in the proceedings of the first temporal GIS workshop (Barrera, Frank and Al-Taha, 1991), and the need for object versions to have an identity is deemed a fundamental temporal issue. The conclusions reached in the workshop were that the identity approach supported by object-oriented databases would be better suited for incorporating time into a GIS rather than the value-based approach of the relational database model. The challenge is to identify the capabilities provided in object orientation for the creation, modification and version maintenance of objects.

Kemp and Kowalczyk (1994) presented the main effort in incorporating a temporal capability within a GIS using the object-oriented paradigm. They employed object-oriented constructs as the tool for providing a flexible and adaptable temporal capability within a GIS. Johnson and Kemp (1995) have also provided a description of the implementation aspects of temporal capabilities within a GIS such as the use of functions (methods) to support spatial and temporal data types and operators.

Table 3.2 GIS version management: an overview.

Level of versioning	Approach	References	Query language	Pros	Cons
Raster cell	Linked lists of attributes for each cell	Langran (1992a)	N/A	Fairly simple approach	Static snapshots
Block	Block sharing in a version-managed B-tree	Newell, Theriault and Easterfield (1994)	Proprietary	Multiple versions of the same object	Designed primarily for long transactions
Relation	Table versioning	Clifford and Ariav (1986)	SQL	Simple and complies with relational algebra	High degree of redundancy
Tuple/object	Time stamps are associated to each tuple	Snodgrass and Ahn (1985)	Extensions to SQL	Efficient storage of versions	Difficulty with temporal joins
Tuple/object	Current tuples are stored in one table, historical tuples in another, linked by pointers	Lum *et al.* (1984)	Extensions to SQL	More efficient storage of versions	Historical queries are computationally intensive
Tuple/object	Time-change objects	Ramachandran (1992)	N/A	Flexible, object-oriented modelling potential	Non-standard database architecture requirements
Tuple/object	A bitemporal element is associated to each object	Worboys (1994)	N/A	Integration of transaction time and valid time	Non-standard database architecture requirements
Attribute	Time stamps are associated to each attribute	Clifford and Ariav (1986)	Extensions to SQL	Low level of redundancy	Non-standard database architecture requirements

3.5 VERSION MANAGEMENT APPROACHES

According to Loomis, 'Versioning is the tracking of the evolution of an object's state through time' (1992, p. 40). Deciding on a version management approach for a spatio-temporal data model involves finding a way to integrate the approach and the model. The version management approach requires being consistent with object-oriented concepts. Table 3.2 describes the main approaches developed for managing versions within databases and their application in a GIS context.

Currently, two main strategies are used to represent multiple versions of an object within an object-oriented database. The first strategy consists of version numbers or time stamps being associated with every attribute or relationship of an object. These attributes or relationships may then be chained, giving them a historical order. The result is a single object and identity (OID) with the versions actually associated with the attributes. This approach could have advantages only in models in which the objects have several attributes as well as changes occurring to only a few attributes.

The second strategy is to track versions at the level of objects rather than attributes. Chaining the old object and any older versions to the new version creates a new version of an object. As a result, a different identifier (OID) is associated with the new version, hence each version has its unique identifier within the version configuration. So far no performance studies have been undertaken on these alternatives. Nevertheless, both alternatives involve the design of operations for handling the dynamic behaviour of versions within the object-oriented data model. Ordering versions within a data model is an important aspect in object orientation. 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.

3.6 CONCLUSIONS

Object orientation is an important related interdisciplinary research domain that can uphold the task of incorporating time in GIS. The fundamental concepts encountered in the object-oriented approach offer useful improvements in functionality, clarity of data modelling and the potential for simplifying future application developments in GIS: 'A common difficulty in…[GIS] application areas is the gulf between the richness of the knowledge structures in the application domains and the relative simplicity of the data model in which these structures can be expressed and manipulated. Object-oriented models have the facilities to express more readily the knowledge structure of the original application' (Worboys, Hearnshaw and Maguire, 1990, p. 370). Therefore, the object-oriented paradigm is viewed as an approach capable of handling the spatio-temporal semantics of Time Geography. The Time Geography framework will be embedded within our spatio-temporal model through the modelling capabilities provided by object orientation. Our spatio-temporal data model is described in the next chapter.