CHAPTER 6

Conceptual Tools for Specifying Spatial Object Representations

Martien Molenaar and *Peter van Oosterom*

6.1 Introduction

The Earth's surface can be considered as a spatio-temporal continuum in which processes of different kinds take place, such as the development of vegetation covers, geologic processes, demographic processes, the development of land use, etc. Each process is the result of interacting forces caused and affected by internal and external factors and will lead to spatial patterns of terrain characteristics which will change with time. Two major classes of such processes can be distinguished:

1. The first class refers to the processes with a field characteristic, i.e., the strength of the interacting forces depends on their positions within the field and the resulting pattern also can be expressed in terms of position-dependent field values;
2. The second class is based on the behavior of (spatially interacting) objects, the resulting pattern of such a process can be expressed by the spatial distribution and the state of these objects.

This chapter will explain some principle characteristics of data models for the representation of object-structured terrain situations in a database environment. The concepts presented here have, to a large extent, been based on Molenaar (1998) and a large part of the text of this chapter is an upgrade and further elaboration of Molenaar (2000). Each represented terrain situation can be considered as a state of a type 2 process at some specified moment and it can be considered a time slice of the spatio-temporal continuum in which such a process has been defined. Time will be dealt with only marginally in this chapter. (See Section 6.4 for further discussion of spatio-temporal GIS.) The emphasis, instead, will be on the representation of spatial objects with their geometric and non-geometric (thematic) aspects. The main aim of this chapter is to make the reader aware of some important decisions or choices that have to be made to formulate models for such spatial object representations.

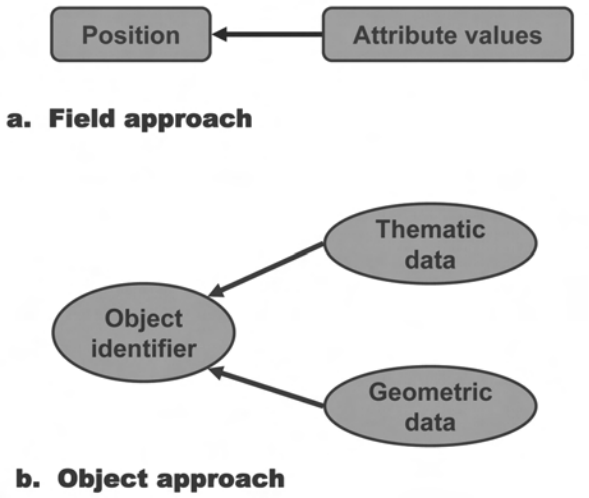

Figure 6-1 Two types of spatial representations (modified from Molenaar 1998).

For most applications, the thematic aspects of a terrain description are of prime importance. This means that data querying and processing will be seen primarily in a thematic perspective. The analysis of the geometric aspects of the data will be secondary and it will depend on the thematic problem formulation.

Two principal structures are used for linking thematic and geometric data in a GIS environment (Peuquet 1990; Laurini and Thompson 1993; Chrisman 1997; Burrough and McDonnell 1998):

1. For the field approach, thematic terrain characteristics are represented by attributes with position-dependent values. The thematic attributes are therefore directly linked to position attributes. This structure has been represented in Figure 6-1a.
2. For the object-structured processes, terrain features or objects are defined, each having a geometric position and shape and several non-geometric characteristics. These objects are represented in an information system by means of an identifier to which the thematic data and the geometric data are linked, as in Figure 6-1b.
   Note: the reader should take care not to confuse the terminology used here with the terminology used for modern techniques in computer science; there are of course relationships between the concepts used in these different fields but there are also important differences in the way the terminology is used.

The representation of a field in a geodatabase requires that the spatial continuum be discretized. This is generally done in the form of points or finite cells often in a regular grid or raster format where the thematic attribute values are evaluated for each point or cell. This is the raster approach. The alternative approach is that the linear structure of terrain features is represented geometrically in the form of chains of points or vertices linked by edges. This is the vector approach.

One should be careful not to equate the field approach with the raster approach or the object approach with vector approach. The concepts of fields and objects refer to the semantics of the represented phenomena, whereas raster and vector data refer to the representation of their geometric aspects (Molenaar 1998). It is very well possible to represent the geometry of field phenomena in a vector structure and to represent the geometry of objects in a raster format.

The remainder of this chapter will deal with the representation of geospatial objects in spatial information systems and is largely based on the presentation of the subject matter in Molenaar (1998). Section 6.2 will elaborate some general characteristics of spatial objects and some of their geometric aspects. Section 6.3 explains several semantic aspects of object modeling:

- Section 6.3.1 discusses the role of thematic object classes and classification hierarchies for organizing the thematic object descriptions.
- Section 6.3.2 describes how aggregation hierarchies can be used to link objects at different levels of complexity.
- Section 6.3.3 explains similarities and differences between object aggregations and object associations.
- Section 6.3.4 focuses on the semantic aspects of three-dimensional object models
- Section 6.3.5 adds the temporal dimension to the spatial objects, resulting in so-called spatio-temporal models
- Section 6.3.6 explains the important role of constraints for refining the semantics in models

Section 6.4 discusses spatial object models in the context of the model phases and involved layers of models. It also gives a short overview of modeling tools such as the Unified Modeling Language (UML) and the Extensible Markup Language (XML), along with standards from the International Standards Organization (ISO) and the Open Geospatial Consortium (OGC). Finally, Section 6.5 explains how object definitions and object descriptions are related to the context in which a spatial database will be used.

6.2 Spatial Objects

In the object-structured approach, the link between the thematic data and the geometric data is made through an object identifier as in Figure 6-1b. The definition of the objects will primarily be made within a thematic or application context: the objects within a cadastral environment will obviously be quite different from the objects in a topographic survey or in a land use analysis at a continental scale. The relevant thematic aspects will be specified within a specific application context. The thematic descriptions of the objects will then be organized in attribute structures which are different for different types of objects or object classes, Figure 6-2.

If geometric information is required, then several decisions have to be made. For each class of objects a decision should be made whether the objects will be represented as point-, line- or area objects, see Figure 6-2 (Molenaar 1998):

- For point objects only the position is stored.
- For line objects the position and shape are stored, "length" is the only size measure given.
- For area objects position and shape will be given, the length of the perimeter and the area are the size measures.

This choice should depend primarily on the role that the objects play in the terrain description and analysis rather then on the actual appearance of the objects or the scale or resolution of the terrain description. A river might be considered as an area object by the authority responsible for the maintenance, whereas that same river might be considered as a line object when it is seen as a part of an hydrological network for flow analysis. A city might be represented as a point object when it is considered as a node in air traffic networks, but the same city will represented as an area object in the case of urban land use mapping.

A decision should be made about the required accuracy of the geometric description of the terrain objects. This concerns the location accuracy, the precision of the shape description and the required geometric detail. Furthermore there is the choice of the geometric format, i.e. whether geometry will be handled in a raster or a vector format (Figure 6-3). This is a

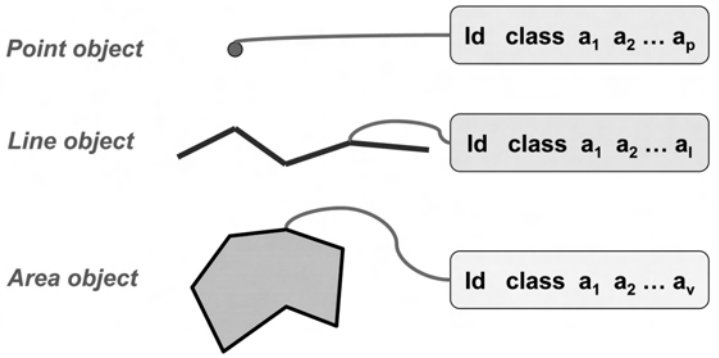

Figure 6-2 Three geometric object types (modified from Molenaar 1998). See included DVD for color version.

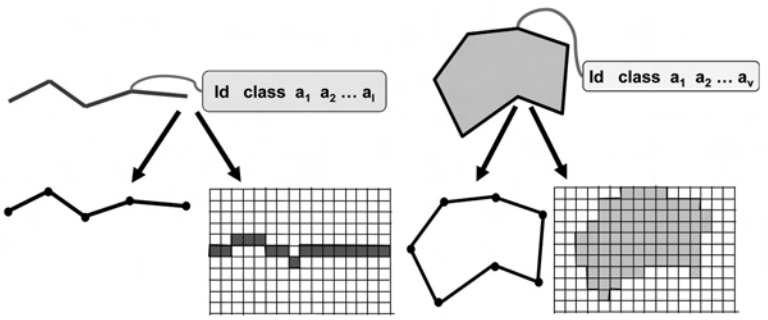

Figure 6-3 Raster and vector representations of spatial objects. See included DVD for color version.

choice for the most convenient format (i.e., it is not a matter of principle) and will depend on the available software and/or on the type of geometric queries and operations that should be performed. Some will be easy to handle in a raster format (like overlaying) whereas others might be easier to handle in a vector format (like many topology-related queries).

Spatial terrain objects are generally part of spatial complexes consisting of many topologically related objects, i.e., consisting of a contiguous set of mutually adjacent area objects intersected by line objects and containing point objects as in Figure 6-4. The representation of the geometric aspects of these objects should therefore be based on a joint common set of geometric elements that contain the topologic information to support the querying and analysis of these spatial relationships (Egenhofer and Herring 1992; De Floriani et al. 1993). The vector structure is very suitable for the implementation of such topologic structures (Molenaar 1998).

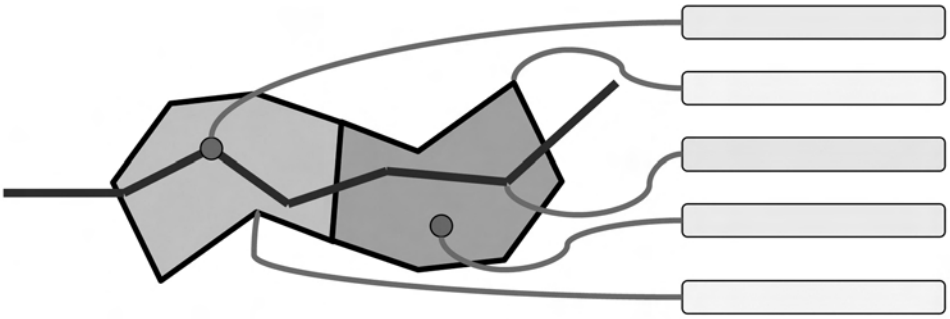

Figure 6-4 Spatial complex consisting of topologically related spatial objects. See included DVD for color version. See included DVD for color version.

Two constraints can be formulated for the specification of spatial objects, especially area objects:

1. A spatial data set, or map, should have no spatial ambiguity. The objects of a particular mapping domain, or thematic field, should be specified so that the area objects do not overlap and they should be spatially disjoint. Each point of the mapped area belongs to only one area object.
2. The spatial data set, or map, should be complete. The objects of a particular mapping domain, or thematic field, should be specified so that the area objects form a complete coverage of the mapped area. There should be no holes in the map and all points of the mapped area belong to an area object.

The combination of these two constraints implies that the area objects of a spatial data set, or map, form a spatial partition of the mapped area.

6.3 Semantic Object Modeling

The previous section already mentioned the fact that terrain objects can be organized in classes according to their thematic aspects. Objects classes define characteristics which their members have in common. This principle has been exploited in computer science in techniques implemented in Object Oriented Programming Languages and Object Oriented Database Management Techniques (Smith and Smith 1977; Cox 1987; Hughes 1991). In computer science, classes define description structures for their member objects and operations that can be applied to them (see also Brodie 1984; Brodie and Ridjanovic 1984). The concepts in computer science have been interpreted and modified by several authors for spatial database applications (Egenhofer and Frank 1989; Kemp 1990; Nyerges 1991; Molenaar 1993, 1998; Fritsch and Anders 1996). Many of them emphasize the adaptation of database techniques to the requirements for operations on spatial data. Some authors adapt the concepts to define semantic data models for describing spatial phenomena. Their

emphasis is not on the definition of database operations but rather on the development of conceptual tools for describing the real world and their activities are considered to be in the domain of geo-information theory rather then in computer science. The discussions in this section also will be along this line (Molenaar 1998).

6.3.1 Terrain Object Classes and Generalization Hierarchies

Suppose that a farm consists of several lots, some of which are used as arable land and others as pasture land. The fact that we make a distinction between these two sorts of fields clearly indicates that they are different because the farmer manages these two sorts of land differently. The farmer needs different types of information for their management and this means dealing with two different object classes. Let the objects be identified by an identification number (id). Let the thematic description for arable land be given by the attributes crop type, sowing date, herbicide, fertilizer and crop type last year. Let the description of pasture land be given by grass type, (monthly) biomass production and fertilizer.

A table can be generated for each class. In Figure 6-5, the columns W1, W2 and W3 represent the thematic attributes of the class of pasture land and the columns A1 through A5 are the thematic attributes of arable land; attribute names are given separately. Each class has its own attribute structure and for every object of a class a value will be assigned to each attribute. These values must fall within the range of the attribute domain, which must be defined prior to the actual assignment of attribute values.

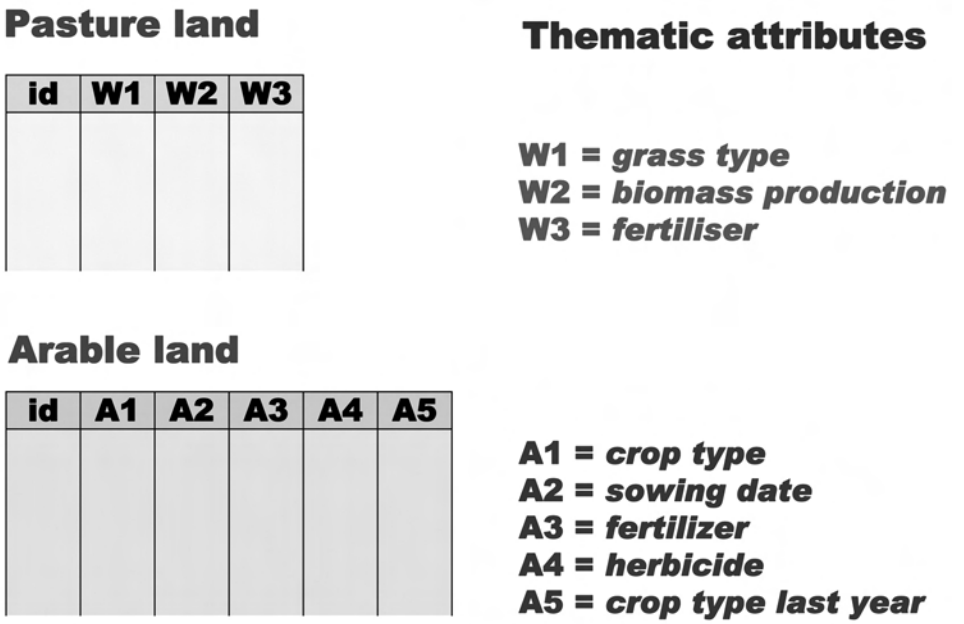

Figure 6-5 Tables for two classes of agricultural land use (modified from Molenaar 1998). See included DVD for color version.

This relationship between objects, classes and attributes has been represented in the diagram in Figure 6-6. Each class has its own unique list of attributes (attribute structure), each object in a class has a list consisting of one value for every attribute. We will assume that each object belongs to only one class, so that the attribute structure of an object is completely determined by the class to which it belongs, i.e., an object inherits the attribute structure of its class. Different classes have different attribute structures, but that is not to say that all attributes are different. We will extend the list of attributes of both classes in our example with area, soil water level and soil type.

Now we have two possibilities for adjusting the table structure of Figure 6-5. We can extend both existing tables with the new attributes (Figure 6-7a) or we can create a new table with these new attributes (Figure 6-7b). The latter implies that all objects that appeared in the two original tables must also appear in the new table. This new table, called farm lot, is a more generalized description of the objects. The distinction between arable land and pasture land is then in fact a further thematic specification of the objects. This is apparent in the fact that per class a more detailed specification of attributes is added to the less specific class of farm lot, as can be seen clearly in the extended lists of attributes for the tables in Figure 6-7a. We speak of farm lot as a generalized class or super-class above the classes of arable land and pasture land. An object that belongs to the class pasture land does not only have the attributes of this class, but also those of the super-class farm lot.

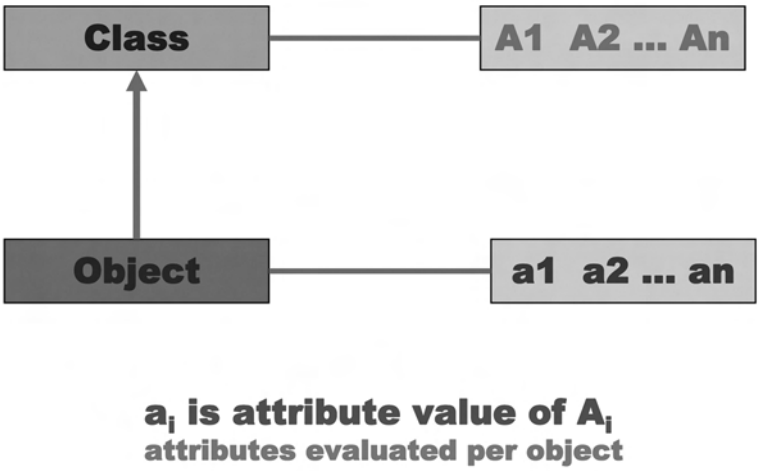

Figure 6-6 Diagram representing the relation between objects, classes and attributes (modified from Molenaar 1998). See included DVD for color version.

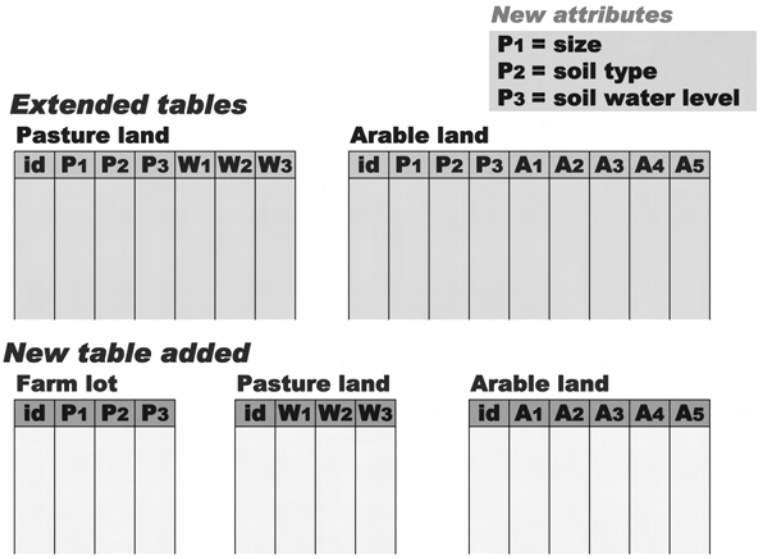

Figure 6-7 Two ways to deal with common attributes for different object classes: A) tables extended for common attributes; B) a separate table for the common attributes (modified from Molenaar 1998). See included DVD for color version.

Figure 6-8 represents a hierarchy of classes created in this way. All classes in a hierarchy can be distinguished by their own unique attribute structure. Within a hierarchic line, these structures are handed down, i.e., objects that belong to a specific class not only have the attributes of that class but also all those of its super-class(es). In a strict hierarchy the relation between a given level and the level above it is always n:1 (many to one), thus a super-class can have many lower-level classes but a class on a lower level belongs only to one super-class on the next higher level. Thus when descending through a hierarchy, we see that at each level an increasingly detailed part of the object's attribute structure is defined. We speak in this case of specialization. At the object level there is no further extension of the attribute list, but the values are assigned to the attributes. When we ascend in a hierarchy, the description of objects becomes less specific and we speak of generalization.

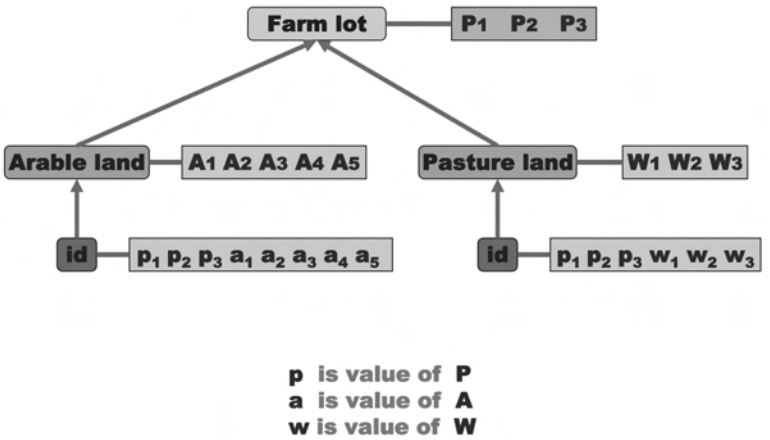

Figure 6-8 Class hierarchy for agricultural objects (modified from Molenaar 1998). See included DVD for color version.

Let U be the collection of all spatial objects represented in a spatial database; we call U the universe of the database. A classification system should be set up so that it is complete and exclusive, i.e., in such a way that all objects of U belong to exactly one class at the lowest level of the system. In that case, the classes at this level form a thematic partition of U (see P1 in Figure 6-9). This implies that within a hierarchy each object also belongs to exactly one class at each higher level of the system so that each level forms a thematic partition of U (see P2 and P3 in Figure 6-9). Then each object of U receives its attribute structure via one and only one inheritance line.

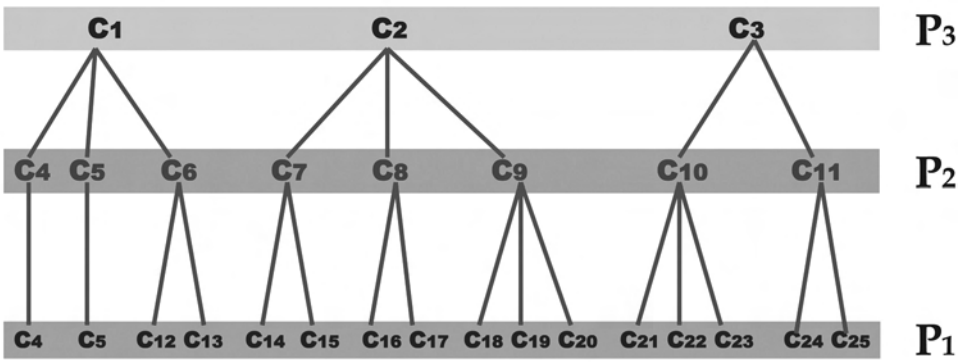

Figure 6-9 An object classification system consisting of several class hierarchies (modified from Molenaar 1998). See included DVD for color version.

Different hierarchical lines, or inheritance lines, can co-exist in one classification system and form one or more trees, as in Figure 6-9. The upwards relation in a classification hierarchy is called an 'ISA' relation. Thus capital ISA city ISA built up area and Amsterdam is an object instance of the object class Capital. The example shows that an ISA relation assigns objects to a class and its super-classes. Classes and super-classes are typified by their attribute structures. A classification system structured in this manner is a manifestation of the concept 'thematic field'. It is represented by a classification system, i.e., a collection of classification hierarchies, with their classes, super-classes and their hierarchical relations. Furthermore, it includes the attribute structure of classes and super-classes, and the attribute domains. A thematic terrain description is the complete set of objects with a list of attribute values per object. In an information system, such a thematic field is always defined in a specific user's context. This context is determined by many factors, including—but not limited to—the mapping discipline, the point in time or era of the mapping, and the scale or aggregation level.

Terrain objects occur at the lowest level in a classification hierarchy and can, therefore, be seen as the elementary objects within the thematic field represented by a given classification system. This implies that the decision whether or not to consider certain objects as elementary must be made within the context of such a thematic field. In other words, this is a context-dependent decision. Objects that are elementary within one thematic field are not necessarily elementary within another.

6.3.2 Aggregation Hierarchies

The fact that the previous subsection dealt with elementary objects implies that there can also be composite objects, or aggregates. These can be defined within the framework of aggregation hierarchies. An aggregation hierarchy describes the way in which composite objects are built up from elementary objects and how these composite objects, in turn, can be combined to form even more complex objects. Figure 6-10 shows an aggregation hierarchy. In the first step from level 1 to level 2, farm fields are combined to form lots. Next these lots are combined with a farmyard to form a farm. In the third step, a number of farms are combined to form an agricultural district.

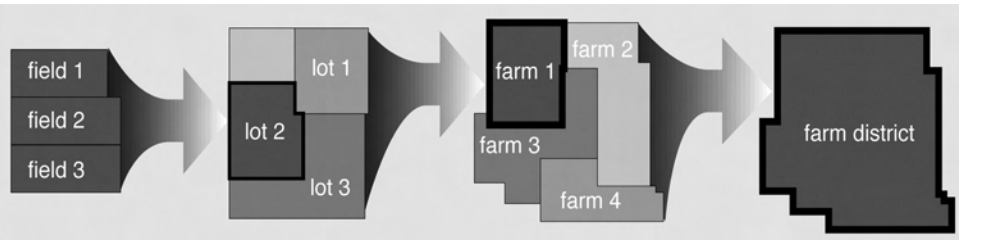

Figure 6-10 The aggregation of agricultural objects (modified from Molenaar 1998). See included DVD for color version.

An aggregation hierarchy has a bottom-up character, in the sense that elementary objects from the lowest level are combined to compose increasingly complex objects as one ascends in the hierarchy. The compound objects inherit the thematic data from their constituent objects and there are generally two types of rules for constructing composite objects. First, there are rules indicating the object classes of which a given compound object can be composed. Second, there are rules indicating which lower level objects can be included in a composite object on the next level. In a GIS, these latter rules are often based on topological relations between objects. The agricultural district in Figure 6-10, for example, is formed from farms that are mutually adjacent and fall within a communal boundary.

This means that aggregation types can be determined by their construction rules. If elementary objects are combined to form a composite object, their attribute values are often aggregated as well. Farm yield is the sum of field yields, and district yield is the sum of farm

yields, as in Figure 6-11. We speak of upwards inheritance in aggregation hierarchies as 'PART OF' relations. For example, 'St. James Park is PART OF Westminster is PART OF London.' The PART OF relations connect groups of objects with a certain aggregate and possibly on a higher level with another even more complex aggregate, and so on.

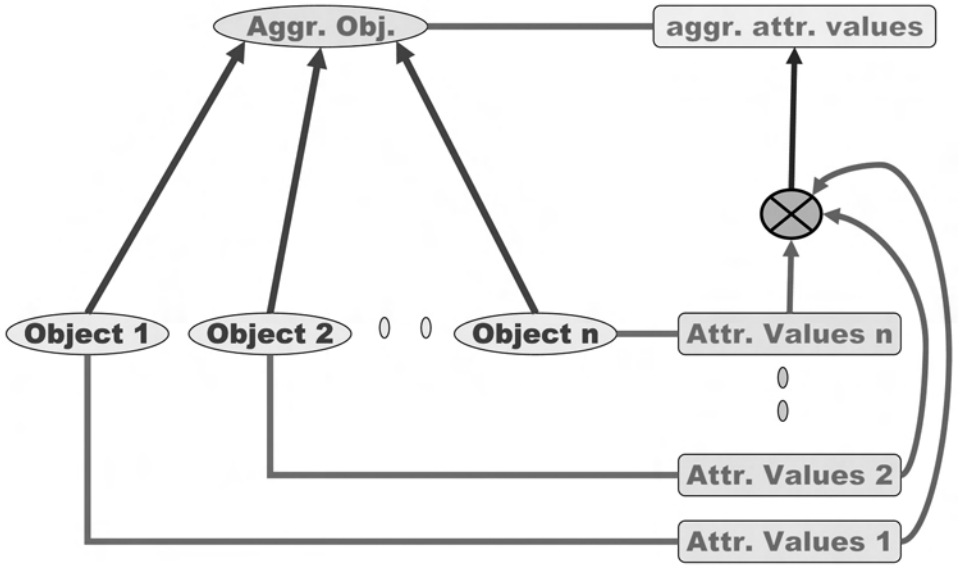

Figure 6-11 Attribute values of the composite object are derived from attribute values of elementary objects (modified from Molenaar 1998). See included DVD for color version.

Another characteristic which distinguishes an aggregation hierarchy from a classification hierarchy is that within a thematic field an elementary object can belong to one and only one class, and thus has only one line of inheritance within a classification structure. It may be part, however, of several different aggregates. A set of aggregate types thus need not be either exclusive or complete. This implies also that not all elementary objects are necessarily a part of an aggregate. For example, hydrological systems and shipping routes are non-exclusive aggregates of waterways. A river can be part of a hydrological system existing of rivers lakes and streams. This same river can also belong to the shipping routes made up of rivers, lakes and canals. However, it does make sense to define aggregates within a hierarchy in such a way that the aggregates of one type are mutually exclusive, i.e., that elementary objects belong to one aggregate of a given type. In this restricted sense the relationships between objects and aggregates are many-to-one, m:1. Thus, a house can only be part of one neighborhood, which can only be part of one municipality. In the same manner, a river can only belong to one hydrological system.

6.3.3 Object Associations

The two types of hierarchies discussed in the previous subsections have clear descriptions. A classification hierarchy has a top down, step-wise introduction of attribute structures for terrain objects. Classes are collections of objects with the same attribute structure. An aggregation hierarchy is defined by the construction rules that describe how the objects on a given level are composed of objects from the next lower level. From the bottom up, the levels have an increasing complexity.

A third way of organizing terrain objects is by more-loosely-defined object associations. The formation of object associations is not bound by a strict set of rules, but objects are grouped on the basis of some common aspects. The relationships that are formed are not necessarily m:1 (many to one) relations, but also may consist of m:n (many to many)

relations. This implies that the associations of one type need not be exclusive. The following example should clarify this principle.

The neighborhoods in Figure 6-12 form associations. The neighborhood of plot 3 is formed by all the other plots that are adjacent to it. In this respect, the composition of an association resembles the composition of an aggregate. The difference, however, is these plots also belong to other neighborhoods, e.g., plot 4 has a neighborhood which overlaps the neighborhood of 3.

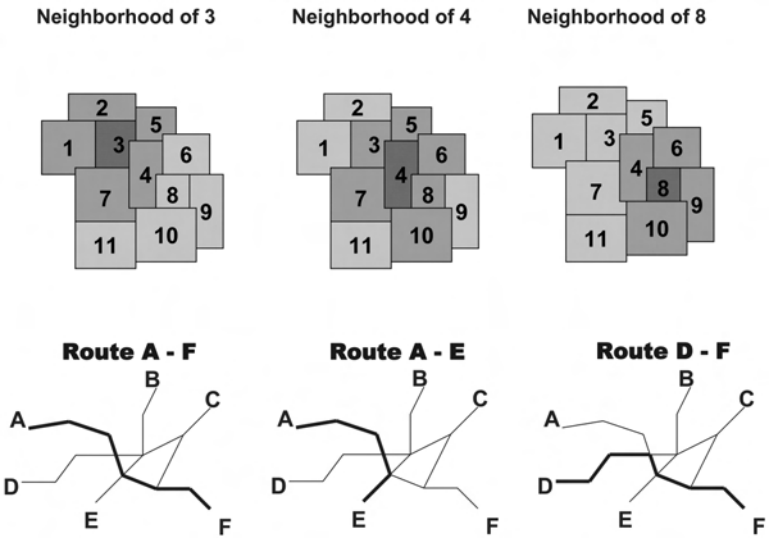

Figure 6-12 Examples of object associations (modified from Molenaar 1998). See included DVD for color version.

The road network in Figure 6-12 also can be regarded as an aggregate; the routes in this network however are another example of associations. The route from A to F consists of different roads or roads segments. The roads (and segments) that are a part of the route from A to F also can be a part of other routes such as from A to E or from D to F. Thus the routes are not mutually exclusive.

These examples show that associations consist of m:n relations. That means that a given object can be a part of several associations of the same type and they do not exclude each other as in the case of aggregations. The relation between an object and an association is called a 'MEMBER OF' relation. For example, 'plot 5 is a MEMBER OF the neighborhood of plot 4'.

6.3.4 Three-dimensional Object Models

Interest in three-dimensional (3D) modeling is growing, both in applications and in science. Typical examples include 3D cadastres (Stoter et al. 2002), telecommunications (Kofler 1998) and urban planning (Cambray 1993). At the same time, geographic information systems (GISs) are changing into integrated architecture in which administrative and spatial data are maintained in one environment in line with the presented modeling theory in Section 6.3.1. It is for this reason that mainstream GIS and database management systems support spatial data types according to the 'Simple Feature Specifications for SQL' described by the OGC. However, these specifications are 2D, as indeed are nearly all the current systems.

The development of 3D applications is mainly due to the growing need for multifunctional use of space by, for example, buildings above roads and railways and bridges and tunnels. As in the two-dimensional (2D) situation, two modeling approaches can be followed (and both are needed) based on:

1. Topological structure where lower level topological primitives (nodes, edges and faces) are used to represent volume, surface, line or point objects (Molenaar 1990; Pilouk 1996).

2. Three dimensional geometric primitives such as polyhedron, sphere, tetrahedron and voxel (Arens et al. 2005).

The absence of real 3D (topological and geometric) primitives in modeling creates major problems. The systems do not recognize 3D spatial objects, because they do not have a 3D primitive to model them. This results in functions that do not work properly; for example, there is no validation for the 3D object (Figure 6-13).

When 3D objects are stored as a set of polygons, no relationship exists between the different polygons that define the object. Besides the fact that validation is impossible and that any set of polygons can be inserted, another disadvantage is that the same coordinates are listed several times (causing inconsistency risks) and there is no information about the outer or inner boundaries (shells) of the polyhedron.

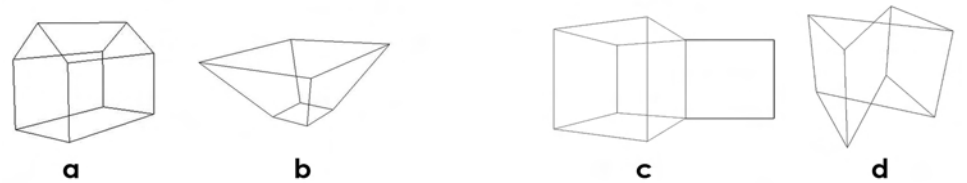

Figure 6-13 Examples of valid (a and b) and invalid (c and d) 3D objects.

6.3.5 Spatio-temporal Models

In the introduction we referred to the Earth's surface as a spatio-temporal continuum in which processes of different kinds take place. This implies that each represented terrain situation should be considered as a state of such a process at some specified moment, i.e., it can be considered a time slice of the spatio-temporal continuum. Clearly the world is not static and changes in objects do occur. Of course these changes should be reflected in the information systems and the models on which these systems are based. Good overviews of handling spatio-temporal data can be found in Langran (1992), Al-Taha et al. (1994), CHOROCHRONOS (1996–2000): A Research Network for Spatiotemporal Database Systems, and Abraham and Roddick (1999). Some papers emphasizing modeling include Worboys (1994), Egenhofer and Golledge (1998), Tryfona and Jensen (1999), van Oosterom et al. (2000) and Peuquet (2002).

When dealing with temporal change of objects, questions such as the following have to be answered (Langran 1992):

- Where and when did a change occur?
- What type of changes occurred?
- What is the rate of change (trend)?
- What is the periodicity of change (if any)?
- Where was this object two years ago?
- How has this area changed over last five years?
- What processes underlie a change?

These questions deal with temporal aspects of spatio-thematic phenomena as represented in the triangle of Figure 6-14a. In order to deal with spatio-temporal data, a system should provide functions such as the following (Langran 1992):

- inventory—complete description at a certain moment in time;
- analysis—explain, exploit, forecast spatial developments and processes;
- updates— supersede outdated spatial and thematic information with new versions;
- quality control—monitor and evaluate new data and check if these are consistent with old data;
- scheduling—identify threshold states which trigger predefined actions; and
- display—generate maps or tables of a temporal process.

A number of basic temporal modeling concepts that pertain to objects will now be introduced. The smallest time unit is called chronon; in a sense this may be compared to the resolution in the spatial domain, such as pixel size. A moment in time is represented with a point on the time line, which runs from left (history) to right (future). There is one very special moment in time and that is now, which is always on the move on the time line. The time line also may branch off to represent different plans/scenarios/predictions (Figure 6-14b). A time interval is the segment on the time line between two moments in time or epochs. A time interval may be used to represent the time a version of an object is valid. The term frequency is used to represent how often certain patterns (of reoccurring events) do happen and may be expressed, for example, in Hertz (cycles per second, hz or 1/sec). Similar to the spatial domain, temporal topology is used in modeling. For moments in time (points on the time line) this is very simple: before, equal or after. The relationships between two intervals of time are more interesting since they can have such topology relationships as disjoint, touch, overlap, include and equal. Even more complex possibilities can be imagined, such as "overlap and end at the same time."

Figure 6-14 Basic spatio-temporal concepts: space-time-theme triangle (a) and timeline (b).

Data granularity related to time is an important aspect when modeling a spatio-temporal dataset as it can range from 'coarse, more redundancy' to 'fine, less redundancy' as the following four examples of data granularity illustrate:

1. Complete universe level or whole data set, e.g., based on aerial photography acquired every 6 years, a new topographic map is produced and the whole data set has the same time stamp,
2. Object class level (that is thematic), e.g., in future topographic maps, certain object classes may be updated more frequently than other object classes such as the road objects updated every 2 years,
3. Object instance level, e.g., individual parcels are updated in the cadastral map on a daily basis such as a complete parcel with all its attributes is renewed, while other objects in the database may remain the same, and
4. Object attribute level; e.g., when measuring every hour the ground water level at a specific point location the other attributes of the ground water measurement object remain the same, but the ground water level attribute is updated very frequently.

For each event there are several moments at which time could be registered:

- when it happened in the real world (real world or user time),
- when it was observed in reality (date photo),
- when it was included in the database (system transaction time),
- when it was last checked in reality,
- the date (time) of the signature/registration/postmark time,
- when an error (in historic data set) was discovered and corrected (two moments),
- when it was last displayed (to user on screen/map).

It is possible to include multiple times in one model and an often-used approached is the bi-temporal model, which includes both user and system time, as discussed below. The changes in a spatial object database (such as in the example of the cadastre, the third example above) are of a discrete type in contrast to more continuous changes in natural phenomena, often represented via the field model (Cheng and Molenaar 1998). There are two types of models for representing this:

1. State orientation—every object is extended with some time attributes (tmin and tmax);
2. Event orientation—store/document the changes (which attribute did change, why, when).

Both state- and event-oriented models are suitable for querying the changes in the past such as 'give changes in map between t1-t2' (Figure 6-15). However, it is not easy to retrieve the situation at any given point in time ('give map at moment t') in the event-oriented approach. Therefore, the state-based approach is used more often and is sometimes mixed with the event approach to document the changes. More information on event-based modeling is given in Chen and Jiang (1998).

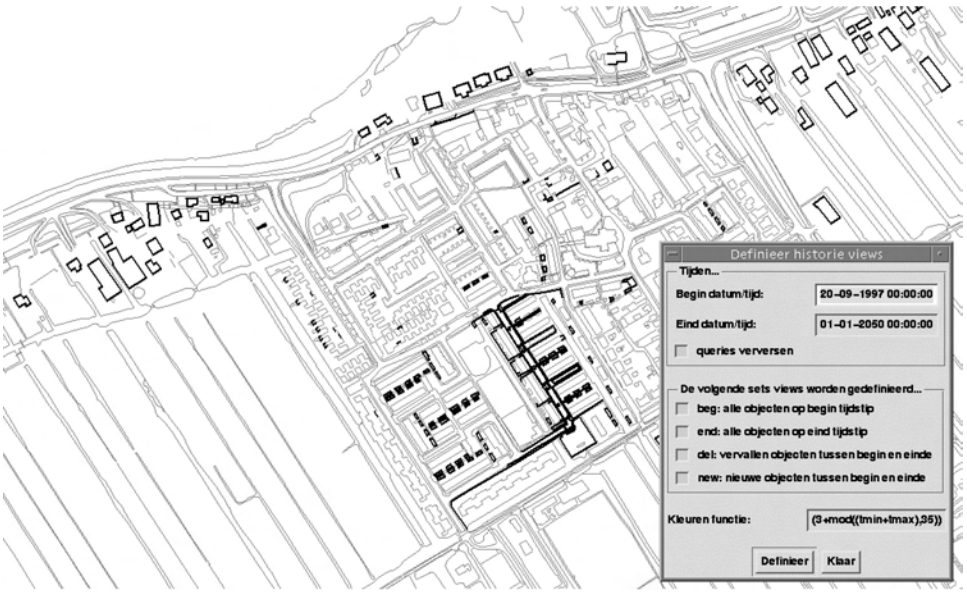

Figure 6-15 Object instance changes in interval. See included DVD for color version.

In state-based temporal modeling (with object instance granularity) a minimal approach is to extend every object with two additional attributes: tmin and tmax as in the Postgres model (Stonebraker and Rowe 1986). The objects are valid from and including tmin and remain valid until and excluding tmax. Current objects get a special tmax value of max_time, indicating they are valid now. These are system times. Furthermore, a model can be extended with the user time attributes valid_tmin and valid_tmax, which would make it a bi-temporal model (Figure 6-16).

When a new object is inserted, the current time is set as the value for tmin, and tmax receives the special value of max_time. When an attribute of an existing object changes, this attribute is not updated, but the complete record, including the object identifier (oid), is copied with the new attribute value. Current time is set as tmax in the old record and as tmin in the new record. This is necessary to be able to reconstruct the correct situation at any given point in history. The unique identifier (key) is the pair (oid, tmin) for every object version in space and time.

In the bi-temporal model, the valid time (user or real world time) and the system time (transaction time) are both maintained. If both time intervals (user and system) are used together, then this results in a time rectangle. Assuming that the system time is after the user time, the lower left (and upper right) is normally below the diagonal of the diagram, which depicts system and user time; see Figure 6-16.

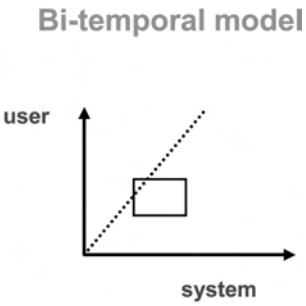

Figure 6-16 Time interval in bi-temporal model. See included DVD for color version.

It should be noted that despite the fact that the state-based, object instance level, temporal model was explained in greater detail, no single temporal model is the best for every situation. The same is true for spatial models. Sometimes a raster model is the best, and sometimes a vector model is the best as with the difference in natural vs. man-made objects. The model that was explained in greater detail was emphasized because it functions relatively well in many situations.

6.3.6 The Role of Constraints in Models

Constraints are important in every GIS modeling process, but until now, constraints have received only ad hoc treatment depending on the application domain and the tools used. In a dynamic context, with constantly changing geo-information, constraints are very relevant. Indeed, any changes arising should adhere to specified constraints, otherwise inconsistencies (data quality errors) will occur. In GIS, constraints are conditions which must always be valid for the model of interest. This chapter argues that constraints should be part of the object class definition, just as with other aspects of that definition, including attributes, methods and relationships. Furthermore, the implementation of constraints (whether at the front-end, database level or communication level) should be driven automatically by these constraint specifications within the model. Figure 6-17, for example, illustrates an edit front-end that includes constraints.

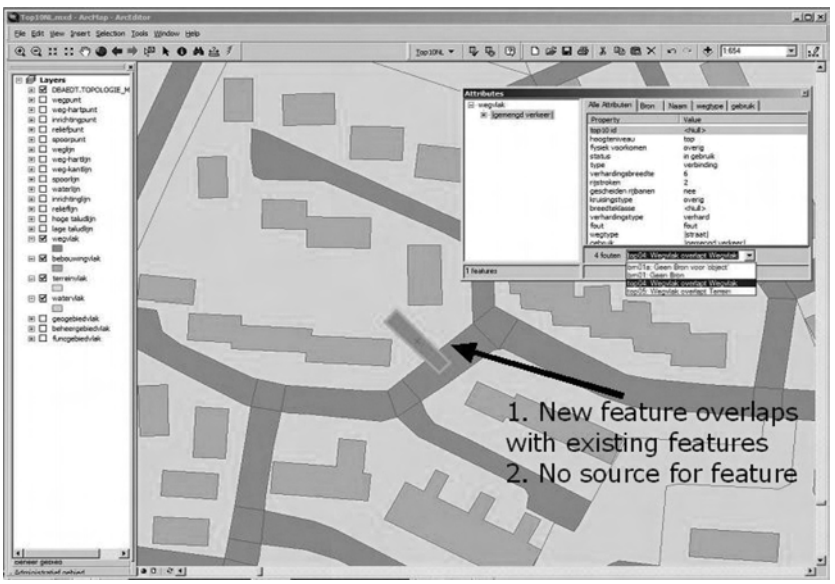

Figure 6-17 Violating a constraint during a topographic update. Source: Topographic Service Kadaster, the Netherlands, from van Oosterom (2006). Copyright © 2006. Dynamic & Mobile GIS: Investigating Changes in Space and Time. ISBN 0-8493-9092-3. Reproduced by permission of Taylor & Francis, a division of Informa plc. See included DVD for color version.

In certain applications some functions—linear programming in spatial decision support systems, survey least squares adjustment, cartographic generalization, editing topologically structured data, etcetera—partially support constraints. However, the constraints are not an integral part of the system and the constraint specification and implementation are often one and the same, and deep in the application's source code. The result is that the constraints are hidden in some subsystems (with other subsystems perhaps unaware of these constraints) and it may be very difficult to maintain the constraints in the event that changes are required. This is true for information systems in general, including GISs, but is especially true for dynamic environments with changing objects, where the support of constraints is required but presents a challenge.

Cockcroft (2004) advocated an integrated approach to handling integrity based on a repository which contains the model together with the constraints. Constraints should be part of the object class definition, similar to other aspects of the definition. The repository is used both by the database and the application as a consistent source of integrity constraints. In order to better understand constraints and their use, it is important to classify them, including their spatial or dimensional aspects. Classification of the different types of (spatial) constraints reveals a complex taxonomy. Cockcroft (1997) presents a 2D taxonomy of (spatial) constraints. The first axis is the static versus transitional (dynamic) distinction and the second axis is the classification into topological, semantic and user constraints. While the transitional aspect of integrity constraints is relevant, in this chapter this is considered to be the 'other side of the same coin.' Van Oosterom (2006), based on four case studies, refines the second axis of Cockcroft's taxonomy by recognizing five sub-axes (or five different criteria) for the classification of integrity constraints:

1. the number of involved objects/classes/instances;
2. the type of properties of objects and relationships between objects involved: topologic (neighborhood or containment), metric (distance or angle between objects), temporal, thematic or mixed;
3. the dimension (related to the previous axis): 2D, 3D or mixed time and space, that is, 4D;
4. the manner of expression: 'never may' (bush never may stand in water) or 'always must' (tree always must be planted in open soil);
5. the nature of the constraint can be 'physically impossible' (tree cannot float in the air) or 'design objective' (bush should be south of tree).

With respect to the first sub-axis of the constraint taxonomy, 'the number of involved objects,' the following cases can be identified:

1. one instance (restrictions on attribute values of a single instance),
2. two instances from the same class (binary relationship),
3. multiple instances of the same class (aggregate),
4. two instances from two different classes (binary relationship) or
5. multiple instances from different classes (aggregate).

Further, the fourth sub-axis, 'the manner of expression,' only has practical value for communicating the constraints between the users. Once the objects and the constraints are formally defined, the expressions 'never may' and 'always must' can be represented by one constraint that is more efficient from an implementation point of view. For example, the constraint 'a tree never may stand in water, or street or house' is equivalent to the constraint 'a tree always must be in a garden or park' under the assumption that there are only five possible ground objects of water, street, house, garden and park.

6.4 Model Phases, Layers and Available Tools

After the above discussion of several aspects of conceptual modeling of spatial objects, other aspects could be added, such as fuzzy or uncertain objects, topology relationships or topology structures (e.g., planar partitions and linear networks) because it is beneficial to realize the bigger picture. Two 'dimensions' of modeling will be discussed below, model phases and model layers. Next, example modeling tools (or standards) will be discussed: UML (including OCL) and XML.

6.4.1 Model Phases

Conceptual modeling is often considered the first level of modeling, giving a model description of the important aspects from the user point of view (semantic aspects). If the involved persons do agree on the conceptual model, then the next step is to translate this into a logical model (Figure 6-18). Before this translation takes place, however, one first has to decide on the type of database platform. Currently the most popular choices are the relational model and the object-relational model. The latter one is an hybrid database form based on the relational model and extended with features of the object-oriented model (such as the ability to add new data types). Other alternatives could be the pure object-oriented database model, network and hierarchical models. The latter two could be considered old-fashioned and do not occur very often anymore. Though one could argue with the advent of XML databases that the hierarchical model is making a comeback. When the conceptual model has been described in a formal manner and a specific logical model is chosen (e.g., object-relational) it is possible to automate the translation to a large extent (e.g., table definitions could be generated automatically). The last step in modeling is then translating the logical model into a physical model. In this step the logical model is used as input and refined with storage and access related measures: exact data types (maximum length of fields), physical clustering, indexing, and perhaps also partitioning, distribution, replication and definition of materialized views.

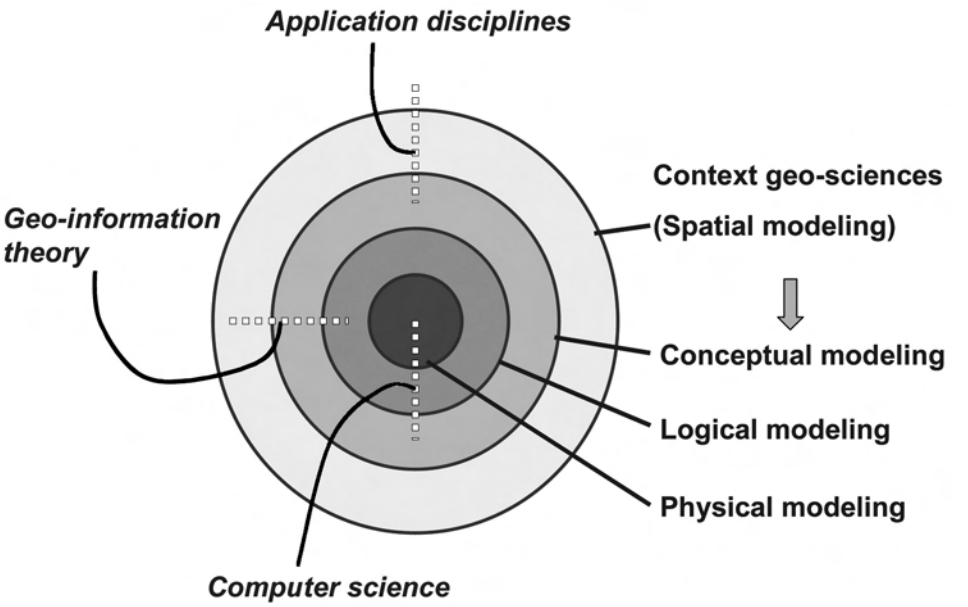

Figure 6-18 Layers (left) and phases (right) of modeling. See included DVD for color version.

6.4.2 Model Layers

Models are not usually built from scratch, but they are built on top of other models. This process can be repeated a number of times and every time the next layer can use the elements defined at the previous layers. Figure 6-18 above shows three such layers: 1) computer science—at this level basic data types and structures are defined including strings, numbers and dates; 2) geo-information theory—how to model spatial-temporal concepts for 2D and 3D geometry and topology primitives; and 3) application disciplines—many different domains such as hydrography, topography, cadastre, soil, geology, transportation, agriculture and environment. Agreeing on these (levels) of models has several advantages: First of all, one does not have to develop everything single-handedly. Second, agreeing on concepts also will enable meaningful exchange of data based on these models. Besides the three mentioned layers, there is at least one other layer that is the final model of a specific application; e.g., the exact model of the Netherlands Topographic Data Set.

6.4.3 Modeling Tools and Standards

When presenting or trying to describe a model, one always faces the question of how to describe this model for domain experts—non-technical end-users or managers who are not modeling experts. This question reappears in every context where models are developed and also applies to the previous sections of this chapter. Textual descriptions alone are difficult to understand as the model structure may not be visible. For the purpose of conveying model structure, all kinds of diagrams have been developed with 'boxes and arrows.' However, the 'boxes and arrows' may have different meanings in different diagrams, making general understanding—even by modeling specialists—difficult. Therefore, the Object Management Group (OMG)) standardized the main types of diagrams and the meaning of 'boxes and arrows' (Booch et al. 2005). The result was the Unified Modeling Language (UML), which is nowadays the state-of-the-art approach for object-oriented modeling (OMG 2002).

UML is a graphic language which gives a wide range of possibilities for representing objects and their relationships. In general, the language can be used for modeling business processes, classes, objects and components, as well as for distribution and deployment modeling. UML consists of diagram elements (e.g., icons, symbols, paths and strings) which can be used in nine different types of diagrams. The most relevant diagram for this discussion of data modeling is the UML class diagram. It provides formalism for describing the objects/classes, with their attributes and behavior, and relationships between these objects such as association, generalization and aggregation. A UML class diagram describes the types of objects and the various kinds of structural relationships that exist among them such as associations (composites, part-whole) and subtypes (specialization-generalization). Furthermore, the UML class diagrams show the attributes and operations of a class and the constraints that apply to the way objects are connected (Booch et al. 2005). UML class diagrams are reasonably well-suited to describe a formal and structured set of concepts, that is, an 'Ontology' (Gruber 1995). Experiences in several different application domains show that it is still not easy to read these diagrams. A good solution for this is the use of 'Literate Modeling,' in which UML diagrams are embedded in text explaining the models. More details and discussion on Literate Modeling, with examples from British Airways, can be found in Arlow et al. (1999).

Having shown the importance of constraints in different applications and having presented a refined taxonomy (in Subsection 6.3.6), the question remains how to specify the constraints. First of all, the specification of the constraints has to be intuitive for the user and the constraints have to be included in the object model. This model should be as formal as possible to enable users to derive constraint implementations within the different subsystems (e.g., edit, store, exchange). Formal modeling is an essential part of every large project; it is also helpful in small and medium-sized projects. Using a formal model facilitates

communicating ideas with other professionals as well as describing clear, unambiguous views on implementation strategies. Despite their potential for formalizing objects and processes, UML class diagrams are typically not sufficiently refined to provide all the relevant aspects of constraints. Constraints are often initially described in natural language and practice has shown that this results in ambiguities. In order to write unambiguous constraints, a non-graphic language is provided within UML for the further modeling of semantics or knowledge frameworks, namely, the Object Constraint Language (OCL) (OMG 2002). When an OCL expression is evaluated, it simply returns a binary value. The state of the system will not change when the evaluation of an OCL expression returns false. The advantage of using OCL is that—as with UML class diagrams—generic tools are available to support OCL and it is not GIS-specific. OCL has been used successfully in the context of GIS, an example being the IntesaGIS project with the GeoUML model specifying the 'core' geographic database for Italy (Belussi et al. 2004). The context of an invariant is specified by the relevant class; e.g., the object class 'parcel' is the context of the constraint 'the area of a parcel is at least 5 m2.' It also is possible within a constraint to use the association between two classes. For example, the constraint that every instance of the object class 'parcel' must have at least one owner could be depicted as an association with the class 'person.' OCL enables one to formally describe expressions and constraints in object-oriented models and other object modeling artifacts. Below are two examples in UML/OCL syntax for the above mentioned constraints (keywords in bold print):

context Parcel **inv** minimalArea:
    self.area > 5

context Parcel **inv** hasOwner:
    self.Owner -> notEmpty()

Figure 6-19 shows the UML class diagram with the objects and the constraints (depicted as associations) used for SALIX-2 (Louwsma 2004). In principle, there is no difference between a 'data model' relationship (association, aggregation, specialization) and a 'data model' integrity relationship constraint. Both are depicted as lines in the UML class diagram. From a high level conceptual (or philosophical) point of view, the difference may be very small. However, normal associations are often indented, in subsequent implementations, to be explicitly stored in one or both directions. The relationship constraints, on the other hand, should not result in such an explicit storage, but in a consistency rule in the implementation environment. In order to distinguish between the two, normal relationships are depicted in black, while integrity relationship constraints are depicted in color. In the diagram notes can be used to explain the constraints on relationships and/or properties. These notes can contain either UML/OCL or natural language text.

Going back to the three different layers of modeling and the relationship to standardized tools, one could state the basic primitives in UML class diagram to form the first (computer science) layer. The second layer is formed by the spatial schema defined by OGC (and ISO 2003). The third layer can be formed by any standardized application domain; for example, the Core Cadastral Domain Model by the FIG (Lemmen et al. 2005). All three are at the conceptual level and use UML for describing the models. At the logical and physical level similar equivalents can be detected within the database and data exchange worlds. For example, the XML schema defines the basic primitives at the first level (XSD 2004). At the second level there is Geography Markup Language (GML) (ISO 2004) and at the third level domain specific XML schemas can be found. Similarly, in the database world these three levels can be found:

1. basic SQL (Date and Darwen 1997);
2. geo-information additions according to the OpenGeospatial 'Simple Feature Specification for SQL' (OGC 2005); and
3. specific database template models for various domains.

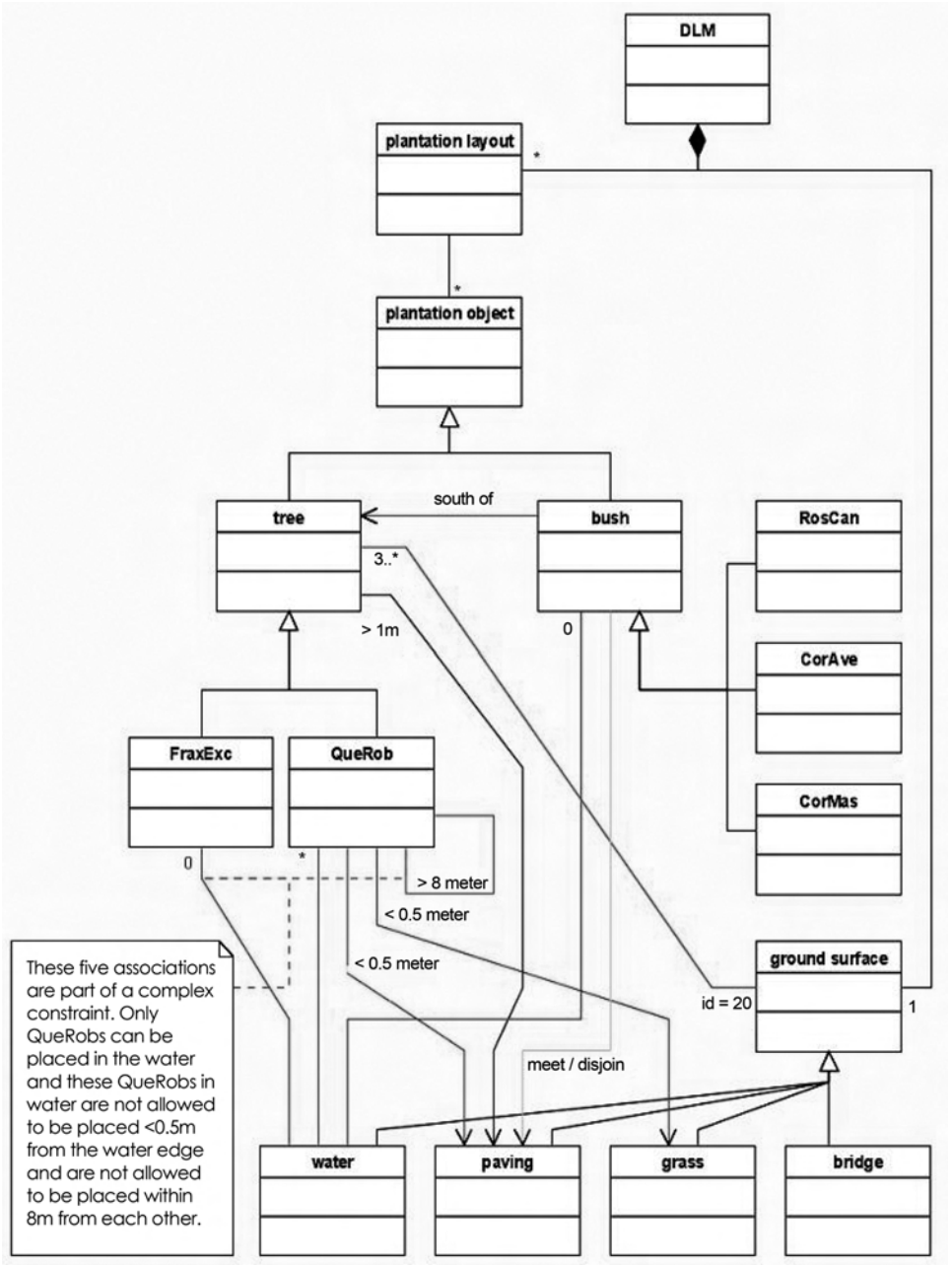

Figure 6-19 Constraints visualized in UML class diagram (Louwsma 2004; van Oosterom 2006). Copyright © 2006 Dynamic & Mobile GIS: Investigating Changes in Space and Time. ISBN 0-8493-9092-3. Reproduced by permission of Taylor & Francis, a division of Informa plc. See included DVD for color version.

6.5 Conclusion: Object Definitions and Context

In Sections 6.2 and 6.3 we described, in general, how a terrain object is represented in an information system via an identifier with associated geometric and thematic data. Section 6.3.1 explained that such objects are meaningful within a certain classification system. Therefore, before a database can be built, the classification structure must be chosen. This choice must be made within a use context with the following characteristics.

The first aspect considers the discipline(s) of the users. One can be dealing, for example, with a soil map or a demographic study or a cadastral system. Each discipline has its own definition of terrain objects, with classes and attributes. These definitions depend not only on the mapping discipline but also on the scale or aggregation level which is used. It makes quite a difference whether certain phenomena (e.g., land use or vegetation) are considered at a local, a regional, a national or a global scale. At each level, different elementary objects are relevant. Furthermore, elementary objects at one level may be aggregates of elementary objects at another level. For example GIS at a municipal level can contain data referring to houses, streets and parks, while a GIS at a national level contains municipalities or built-up areas.

Another aspect of the use context concerns the type of use that is to be made of the data. It makes quite a difference if data are to be used for management purposes or for the analysis of a terrain situation or for processes such as planning and design activities. All of these activities have their own standards for data and terrain descriptions, but that is not to say that there is no overlap.

A final aspect is the point in time in which the terrain description is made. In many cases the value or relevancy of information depends on the era. We can see from an agricultural point of view that the need for information about soils has changed in the course of time. Whereas the major interest used to be in the suitability of soils for certain crops, nowadays there is more interest in the capacity to bind certain chemical elements, with an eye on environmental effects. In cadastres the original task was to collect and store data for raising land tax and/or for the protection of owners' titles, but presently there is an increasing request for economic data such as the dynamics of real estate prices and the number of transactions and mortgages. An operational definition of 'use context' has been given in Bishr (1997); this definition has been based on a formal data schema of Molenaar (1998), specifying that geometric object descriptions should be related to the hierarchical classification models, as in Subsection 6.3.1. The definition of context by Bishr (1997) is, in fact, a meta model describing the semantics of the spatial data model which have been specified for a particular application. The relevance of data always depends on the context in which the data will be used.

If the data are modeled with the concepts presented in this chapter, then a context will be expressed through the semantic definitions of the objects and their actual descriptive structures. This includes, in such a context, the elementary objects with their classes at the different hierarchical levels. Several class hierarchies also may co-exist, i.e., the collection of classes may form one or more trees. If the classes of these hierarchies are defined so that at each level each object of the database is a member of exactly one class, then they form a thematic partition per level. This implies that each object inherits its attribute structure through exactly one inheritance line of such a system. A classification system structured in this manner is a representation of a 'thematic field.' It is characterized by a hierarchical classification system of classes and super-classes with their hierarchical relations. Further characteristics include the attribute structure of classes and super-classes, and the attribute domains. A thematic terrain description is the complete set of objects with a list of attribute values per object. In an information system, such a thematic field is always defined in a specific user's context.

Furthermore, the choice of geometric type must be made for these objects. This again depends on the role the objects are to play in a terrain description. A river can be regarded as a line object in a hydrological database, while the same river can be handled as an area

object in the database of an organization that manages waterways. In the same way, a city can be seen as an area object in a database for demographic studies, whereas the same city appears as a point object, or a node, in a database for continental transport lines. Thus the decision which geometric aspects of a given class of terrain objects are relevant (i.e., the choice of treating objects as points, lines or areas) always depends on the user's context. This implies that the choice of which objects should be regarded as elementary with their relevant thematic and geometric characteristics also depends on the user's context.

Generally the definition and identification of elementary objects follows their thematic specification as expressed through the classification system. The specifications of a spatial data set are unambiguous if two constraints are fulfilled:

1. The classification system is designed so that the classes at each hierarchic level form a thematic partition of the universe of the spatial data base; and
2. The objects have been specified so that they collectively form a spatial partition of the mapped area (according to Section 6.2).

In that case the thematic object classes generate a spatial partition so that there are no spatial overlaps of thematic classes, and therefore there is, in principle, no ambiguity in the thematic classification of areas.

Within such a context, decisions also must be made as to which object aggregates and associations are relevant. Although not necessarily explicitly stored in a database, the decisions may be implied in the form of generic models—rules and procedures to generate these aggregates and associations. Together these choices and decisions define spatial object representations that characterize the spatial patterns of the Earth's surface that, in turn, reflect the interaction of internal and external processes.

References

Abraham, T. and J. F. Roddick. 1999. Survey of spatio-temporal databases. *GeoInformatica* 3: 61–99.

Al-Taha, K. K., R. T. Snodgrass and M. D. Soo. 1994. Bibliography on spatiotemporal databases. *International Journal of Geographical Information Systems* 8:95–103.

Arens, C., J. Stoter and P. van Oosterom. 2005. Modelling 3D spatial objects in a geo-DBMS using a 3D primitive. *Computers & Geosciences* 31 (2):165–177.

Arlow, J., W. Emmerich and J. Quinn. 1999. Literate modelling - capturing business knowledge with the 'UML.' Pages 189–199 in *The Unified Modeling Language: <<UML>>'98: Beyond the Notation*. Edited by Jean Bézivin and Pierre-Alain Muller. First International Workshop, Mulhouse, France, June 3-4, 1998, Selected Papers. Lecture Notes in Computer Science vol. 1618. Berlin: Springer.

Belussi, A., M. Negri and G. Pelagatti. 2004. GeoUML: A geographic conceptual model defined through specialization of ISO TC211 standards. In *Proceedings 10th EC GI & GIS Workshop—ESDI: State of the Art*, Warsaw, Poland, 23-25 June 2004. Ispra, Italy: Institute for Environment and Sustainability, JRC, European Commission <http://www.ec-gis.org/Workshops/10ec-gis/papers/> Accessed 31 January 2007.

Bishr, Y. 1997. *Semantic Aspects of Interoperable GIS*. ITC Publication Series, No. 56. Enschede, The Netherlands. 154 pp.

Booch, G., J. Rumbaugh and I. Jacobson. 2005. *The Unified Modeling Language User Guide,* 2nd edition. Boston: Addison-Wesley Technology Series.

Brodie, M. L. 1984. On the development of data models. In *On Conceptual Modeling*, edited by M. L. Brodie, J. Mylopoulos and J. W. Schmidt, 19–47. Berlin: Springer Verlag.

Brodie, M. L. and D. Ridjanovic. 1984. On the design and specification data base transactions. In *On Conceptual Modeling*, edited by M. L. Brodie, J. Mylopoulos and J. W. Schmidt, 277–306. Berlin: Springer Verlag.

Burrough, P. A. and R. A. McDonnell. 1998. *Principles of Geographical Information Systems*. Oxford: Oxford University Press.

Cambray, B. 1993. Three-dimensional modeling in a geographical database. Pages 338–347 in *Proceedings of AUTO CARTO 11: Eleventh International Symposium on Computer-Assisted Cartography*, held in Minneapolis, Minn. October 1993. Bethesda, Md.: ASPRS.

Chen, J. and J. Jiang. 1998. An event-based approach to spatio-temporal data modelling in land subdivision systems. *GeoInformatica* 2:387–402.

Cheng, T., and M. Molenaar. 1998. A process-oriented spatio-temporal data model to support physical environmental modelling. Pages 418–430 in *Proceedings of the 8th International Symposium on Spatial Data Handling*, held in Vancouver, British Columbia. Edited by T. K. Poiker and N. Chrisman. Burnaby, British Columbia: International Geographical Union.

CHOROCHRONOS. 1996–2000. A Research Network for Spatiotemporal Database Systems <http://www.dbnet.ece.ntua.gr/~choros> Accessed 10 January 2007.

Chrisman, N. R. 1997. *Exploring Geographic Information Systems*. New York: John Wiley & Sons.

Cockcroft, S. 1997. A taxonomy of spatial data integrity constraints. *GeoInformatica* 1 (4):327–343.

———. 2004. The design and implementation of a repository for the management of spatial data integrity constraints. *GeoInformatica* 8 (1):49–69.

Cox, B. J. 1987. *Object Oriented Programming*. Reading, Mass.: Addison-Wesley.

Date, C. J. and H. Darwen. 1997. *A Guide to the SQL Standard*, 4th edition. Reading, Mass.: Addison-Wesley.

De Floriani, L., P. Marzano and E. Puppo. 1993. Spatial queries and data models. In *Spatial Information Theory, a Theoretical Basis for GIS*, edited by A. U. Frank and I. Campari, 113-138. Berlin: Springer-Verlag.

Egenhofer, M. J. and A. U. Frank. 1989. Object oriented modeling in GIS: Inheritance and propagation. Pages 588–598 in *AUTO CARTO 9 Proceedings. Ninth International Symposium on Computer-Assisted Cartography*, Baltimore, Md., 2-7 April 1989. Falls Church, Va.: ASPRS.

Egenhofer, M. J. and R. G. Golledge, editors. 1998. *Spatial and Temporal Reasoning in Geographic Information Systems*. Oxford: Oxford University Press. 276 pp.

Egenhofer, M. J. and J. R. Herring. 1992. *Categorizing Binary Topological Relationships Between Regions, Lines, and Points in Geographic Databases*. Technical report, Department of Surveying Engineering. Orono: University of Maine.

Fritsch, D. and K. H. Anders. 1996. Objectorientierte Konzepte in Geo-Informationssystemen. *Geo-Informations-Systeme* 9 (2):2–14.

Gruber, T. R. 1995. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies* 43 (5):907–928.

Hughes, J. G. 1991. Object-oriented Databases. New York: Prentice Hall.

ISO. 2003. ISO/TC 211, ISO 19107:2003. *Geographic Information—Spatial Schema*. Geneva, Switzerland: ISO.

———. 2004. ISO/TC211, ISO 19136 *Geographic Information—Geography Markup Language*. Geneva, Switzerland: ISO.

Kemp, Z. 1990. An object-oriented model for spatial data. Pages 659–668 in *Proceedings of the 4th International Symposium on Spatial Data Handling, Zurich, July 1990*. Edited by K. Brassel and H. Kishimoto. Zürich: University of Zürich.

Kofler, M. 1998. *R-trees for the Visualizing and Organizing Large 3D GIS Databases*. Ph.D. Dissertation. Graz, Austria: Technical University. 131 pp.

Langran, G. 1992. *Time in Geographic Information Systems*. London: Taylor & Francis.

Laurini, R. and D. Thompson. 1993. *Fundamentals of Spatial Information Systems*. London: Academic Press.

Lemmen, C., P. van Oosterom, J. Zevenbergen, W. Quak and P. van der Molen. 2005. Further progress in the development of the core cadastral domain model. Proceedings of

'From Pharaohs to Geoinformatics', FIG Working Week 2005 and GSDI-8, Cairo, Egypt, April 16-21, 2005. Frederiksberg, Denmark: International Federation of Surveyors (FIG) <http://www.fig.net/pub/cairo/papers/ts_11/ts11_01_lemmen_etal.pdf> Accessed 31 January 2007.

Louwsma, J. H. 2004. *Constraints in geo-information models; Applied to geo-VR in landscape architecture*. MSc Thesis. Geodetic Engineering, Delft University of Technology, The Netherlands. 104 pp.

Molenaar, M. 1990. A formal data structure for 3D vector maps. Pages 770–781 in *Proceedings EGIS*, Amsterdam, April 10-13, 1990, vol. 2. Edited by J. Harts, H.F.L. Ottens and H. J. Scholten. Utrecht, The Netherlands: EGIS Foundation.

———. 1993. Object hierarchies and uncertainty in GIS or Why is standardisation so difficult. *Geo-Informations-Systeme*. 6:22–28.

———. 1998. *An Introduction to the Theory of Spatial Object Modelling for GIS*. London: Taylor & Francis. 229 pp.

———. 2000. Conceptual tools for specifying geospatial descriptions. In *Geospatial Data Infrastructure – Concepts, Cases and Good Practice*, edited by R. Groot and J. Mclaughlin, 151–173. Oxford: Oxford University Press.

Nyerges, T. L. 1991. Representing geographical meaning. In *Map Generalisation: Making Rules for Knowledge Representation*, edited by B. P. Buttenfield and R. B. McMaster, 59–85. London: Longman.

OGC. 2005. OpenGeospatial Consortium, Inc. *OpenGIS® Implementation Specification for Geographic Information - Simple Feature Access - Part 2: SQL Option*, Version: 1.1.0. OpenGIS Project Document OGC 05-134, 22 November 2005.

OMG. 2002. Object Management Group. *Unified Modeling Language Specification* (Action Semantics), UML 1.4 with action semantics. January 2002.

Peuquet, D. J. 1990. A conceptual framework and comparison of spatial data models. In *Introductory Readings in GIS*, edited by D. J. Peuquet and D. F. Marble, 250–285. London: Taylor & Francis.

———. 2002. *Representations of Space and Time*. New York: Guilford Press. 380 pp.

Pilouk, M. 1996. *Integrated Modelling for 3D GIS*. Ph.D. Dissertation, ITC, The Netherlands. 200 pp.

Smith, J. M. and D.C.P. Smith. 1977. Database abstractions: Aggregation and generalization. *ACM Transactions on Database Systems* 2:105-133.

Stonebraker, M. and L. A. Rowe. 1986. The design of POSTGRES. Pages 340–355 in *Proceedings of the 1986 ACM SIGMOD International Conference on Management of Data*, May 28-30, 1986, Washington, DC. Edited by Carlo Zaniolo. New York: ACM Press.

Stoter, J. E., M. A. Salzmann, P.J.M. Van Oosterom and P. Van der Molen. 2002. Towards a 3D cadastre. *Proceedings FIG ACSM/ASPRS*, Washington DC, USA. Frederiksberg, Denmark and Bethesda, Md.: FIG/ACSM/ASPRS. Digital CD. 12pp.

Tryfona, N. and C. S. Jensen. 1999. Conceptual data modelling for spatiotemporal applications. *GeoInformatica* 3:245–268.

van Oosterom, P.J.M., B. Maessen and C. W. Quak. 2000. Generic query tool for spatio-temporal data. *International Journal of Geographical Information Science* 16 (8):713–748.

van Oosterom, P.J.M. 2006. Constraints in spatial data models, in a dynamic context. Chapter 4 in *Dynamic & Mobile GIS: Investigating Changes in Space and Time*, edited by J. Drummond, R. Billen, D. Forrest and E. João, 104–137. London: Taylor & Francis.

Worboys, M. F. 1994. Unifying the spatial and temporal components of geographical information. Pages 505–517 in *Advances in GIS: Proceedings of the 6th Symposium on Spatial Data Handling*, Edinburgh, Scotland, September 1994. Edited by T. Waugh and R. Healey. London: Taylor & Francis.

XSD. 2004. *XML Schema*. <http://www.w3.org/XML/Schema> Accessed 15 January 2007.