TECHNICAL REPORT CH-98-10

A COMPONENT-BASED CONCEPTUAL MODEL FOR SPATIOTEMPORAL DESIGN

NECTARIA TRYFONA and CHRISTIAN S. JENSEN

DECEMBER 1998

CHOROCHRONOS: TMR Research Network Project, No ERBFMRXCT960056
Contact Person: Prof. Timoleon Sellis, Dept. of Electrical and Computer Engineering, National Technical University of Athens, Zografou 15773, GR Tel: +30-1-7721601, FAX: +30-1-7721659, timos@cs.ntua.gr

A Component-Based Conceptual Model for Spatiotemporal Applications Design[1]

Nectaria Tryfona Christian S. Jensen
Department of Computer Science
Aalborg University
Fredrik Bajers Vej 7E, DK-9220 Aalborg Øst, Denmark
{tryfona,csj}@cs.auc.dk

Abstract

Conceptual data modeling for complex applications, such as multimedia and spatiotemporal applications, often results in large, complicated and difficult-to-comprehend diagrams. One reason for this is that these diagrams frequently involve repetition of autonomous, semantically meaningful parts that capture similar situations and characteristics. By recognizing such parts and treating them as *units*, it is possible to simplify the diagrams, as well as the conceptual modeling process. We propose to capture autonomous and semantically meaningful excerpts of diagrams that occur frequently as *modeling patterns*. Specifically, the paper concerns modeling patterns for conceptual design of spatiotemporal databases. Based on requirements drawn from real applications, it presents a set of modeling patterns that capture spatial, temporal, and spatiotemporal aspects. To facilitate the conceptual design process, these patterns are abbreviated by corresponding spatial, temporal, and spatiotemporal *pattern abstractions*, termed *components*. The result is more elegant and less-detailed diagrams that are easier to comprehend, but yet semantically rich. The Entity-Relationship model serves as the context for this study. An extensive example from a real cadastral application illustrates the benefits of using a component-based conceptual model.

Keywords: conceptual modeling, spatial and temporal aspects, spatiotemporal applications, data modeling patterns, components, CASE tool.

1. Introduction

Conceptual data modeling is an indispensable part of information systems design and development. It is also a mature field with broadly accepted notions, methods, models, and tools. Its main objective is to represent an application domain in a manner not requiring any computer metaphors and that is understandable to the user and is complete, so that this representation can be translated into a corresponding logical schema without any further user input. Conceptual modeling is done using one of several variants of the Entity-Relationship (ER) model (Chen, 1976), or one of the more versatile semantic models such as IFO (Abiteboul and Hull, 1987) and SDM (Hull and King, 1987). Alternatively, object-oriented software design tools, like OMT (Rumbaugh, et. al., 1991), provide similar functionality.

The result of conceptual data modeling is a semantic (or conceptual) schema, i.e., a diagram using the notation of the chosen conceptual data model, that captures the desired aspects of the reality modeled at a high level of abstraction. In complex and semantically rich application domains, these schemas are often large and complicated. Some of this complexity is unnecessary: a closer look reveals that considerable parts of the schemas are often repeated to address semantically similar situations and characteristics. For example, in multimedia applications, the characteristics of audio and video are attached to any of the object-actors in a multimedia scene. In a banking system, similar temporal aspects of many different types of objects, such as accounts and loans, are captured.

[1] This research was supported in part by the Danish Technical Research Council through grant 9700780, the Danish Natural Science Research Council through grant 9400911, and the CHOROCHRONOS project, funded by the European Commission DG XII Science, Research and Development, as a Network Activity of the Training and Mobility of Researchers Program, contract no. FMRX-CT96-0056.

Although considerable research on extensions of conceptual models to capture semantics of specialized and complicated environments *does* exist, the issue of recognizing repeated diagrammatic parts and handling them in a more elegant way in order to facilitate the design process for non-standard application areas has not yet been addressed satisfactorily.

This paper focuses on *modeling patterns* for spatiotemporal applications, applications involving time-varying, geo-referenced information. A modeling pattern is a part of the conceptual schema, which appears repeatedly in a schema and describes semantically similar situations and characteristics. When designing a spatiotemporal conceptual schema, it is often necessary to repeat parts of the schema in order to capture the spatial, temporal, and spatiotemporal aspects of the modeled concepts. For example, all geographic objects have positions in space, or geometry, and spatiotemporal objects have combined spatial and temporal aspects.

The main objective of this work is, based on explicit spatiotemporal requirements, to detect the modeling patterns in a conceptual spatiotemporal schema and to propose an elegant way to handle them, in order to facilitate the conceptual design process. We propose the replacement of patterns with a small set of spatial, temporal and spatiotemporal *pattern abstractions*, which we call *components*, resulting into a less-detailed, but equally semantically rich, conceptual schema. These components can be built out of the constructs of *any* conceptual model based on the approach presented in this paper, and can serve as "plug-in's" to that model. We describe the patterns and components in the following way.

a. First, we present fundamental requirements posed by spatiotemporal applications.

b. Then, for each requirement, a corresponding conceptual schema, or pattern, is developed that addresses the requirement.

c. Next, for each pattern, we present the *component* that abbreviates the pattern.

This way, we obtain two levels in conceptual diagrams:

- one detailed enough for the designer to explain all the information that is needed to describe an application and to be considered for the translation to the logical schema, and
- an abstract one, using components, that is easier for the user to follow, free of details, and less complicated, but which still captures the desired spatiotemporal aspects of the modeled reality.

The result is a component-based conceptual model with an associated library of spatiotemporal components, which is tailored to the needs of time-varying, geo-referenced applications. This is part of a larger effort to design a spatiotemporal, computer-aided software engineering (CASE) tool that supports the transition of conceptual schemas to the logical and the implementation levels. The library of components helps the designer (and, finally, the end-user) to, concisely and cleanly, capture the desired spatiotemporal aspects of the modeled reality. The ER model is used as a prototypical context for the study. Any other conceptual model would also do for this purpose.

We should point out that we do not extend the ER model to capture spatiotemporal requirements; several works exist in this area (Claramunt, et. al., 1997) (Story and Worboys, 1995) (Tryfona and Jensen, 1998). Here, we use the ER model only as a vehicle to capture and formulate spatial, temporal, and spatiotemporal patterns.

We are not aware of any other work in this direction. (Hay, 1996) proposes generic conceptual (sub-) diagrams for specific application areas (e.g., accounting) which are to be “instantiated” when applied in specific diagrams (e.g., concerning bank accounting). There also are some works in classification and reusability of conceptual schemas. (Teory et al., 1989) present a set of operators for clustering of ER schemas, leading to layers of abstraction that support different user views. (Castano et al., 1992), (Castano and DeAntonellis, 1994), and (Delcambre and Lanngston,

1996) address the issue of recognizing patterns in conceptual schemas in order to facilitate the design process. These patterns refer to the structual part of the schema (e.g., entities with the same number of attributes represented in the same diagram) as well as the textual part (e.g., entities with the same name). Finally, in (Claramunt et al., 1997) the main focus is on the analysis of spatiotemporal processes as patterns, rather that on the design of modeling patterns from model constructs.

The rest of the paper is organized as follows. Section 2 gives a general overview of modeling patterns and the benefits of using them. A comparison between the design patterns approach, which very often on used in object-oriented modeling and our modeling patterns-and-components approach is given in Section 2.2, in order to clarify the relation between the two. Section 3 presents spatial, temporal, and spatiotemporal requirements. Then, in Section 4, modeling patterns and corresponding components are given, that addresse each requirement, resulting in a library of ER spatiotemporal components. Two approaches for their use are offered: a top-down and a bottom-up. Section 5 gives a detailed example from a cadastral application modeled in the component-based ER and in the conventional ER model. Finally, Section 6 summarizes and offers directions for further research.

2. Modeling Patterns

2.1 Modeling Pattern and Components in Conceptual Diagrams

A specific notion of pattern that originally stems from architecture has found use in software engineering (e.g., see (Gamma et al., 1994)). In object-oriented software design, programmers adapt parts of already developed program code to create new code; this way, solutions to new problems are developed cost-efficiently (Pree, 1994). Stated briefly, these reusable code fragments are termed design patterns. Design patterns have also been studied in the context of geographic information systems (Gordillo, et. al., 1997).

In the same way that code fragments may form reusable patterns, fragments of conceptual diagrams are often used repeatedly to express the semantically same situation. We term such recurring fragments *instantiations of modeling patterns*.

Modeling patterns are *generic, autonomous,* and *semantically meaningful excerpts* of conceptual database schemas that capture similar semantic aspects of the application domain. When specific names are given to the modeling constructs in a pattern, the pattern is instantiated. For example, it is fundamental in spatiotemporal applications for entities to have spatial extents. In the ER model, a set of entities with spatial extents may be captured using a combination of modeling constructs; and different entity sets with spatial extents may be captured using the same combinations of constructs–it is only the names used with the constructs that differ. As a result, we form a pattern for entities with spatial extents and perceive different sets of entities with spatial extents as instantiations of this pattern.

Modeling patterns have two characteristics: (a) they are *generic sub-diagrams*, with multiple instantiations *in a single* conceptual *schema* or *across* schemas, and (b) they are *composite diagrams*, meaning they are built from atomic modeling constructs of a conceptual model (such as entity sets, relationship sets, or attributes). Design patterns differ from modeling patterns in that, the latter address only structural aspects at the conceptual level, while the former concern also behavior and dynamic properties of program code.

Figure 1 illustrates the idea of the use of modeling patterns: the shaded areas show repeated pattern instantiations in an ER diagram. At the next, less detailed level (level 2), these patterns are replaced by *pattern abstractions*, which we term simply *components*.

In the same way, components at level 2, together with other components and atomic modeling constructs, may form higher-level components. In this work, we consider only one level of components, i.e., those built from atomic modeling constructs of the underlying conceptual model. Components (at a coarser level) are connected to the rest of the conceptual diagram in the same way as regular ER modeling constructs such as entity types and relationship types are connected.

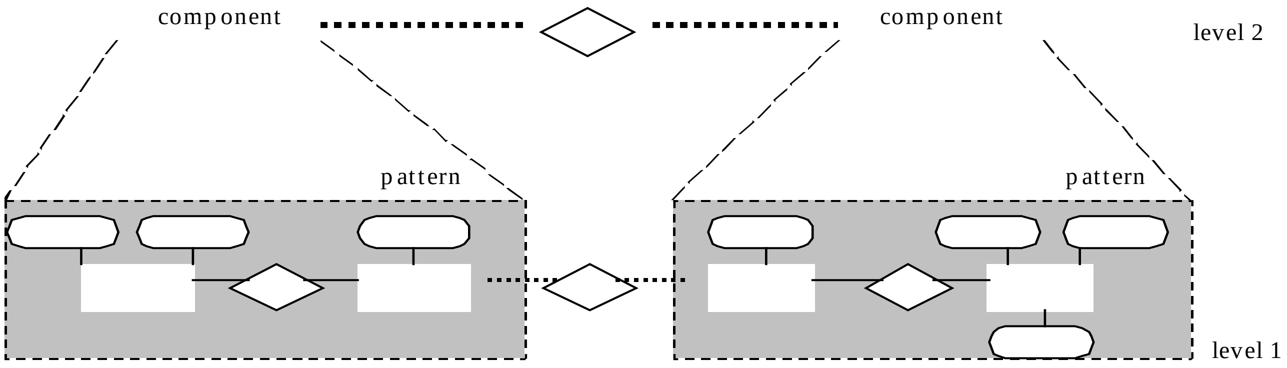


Figure 1: Modeling patterns forming components in a conceptual schema.

It is important to emphasize that patterns capture *semantically similar aspects* of the modeled reality. This is why we take our outset in spatiotemporal requirements to conceptual data modeling. Syntactic similarities in diagrams, such as occurrences in a diagram of different entity types having the same specific number of attributes, are outside the interest of this work. Furthermore, we do not use guidelines to cluster parts of a conceptual schema, or operators to replace patterns with components.

The anticipated advantages of using modeling patterns are as follows.

- Organization of the conceptual database schema into levels of abstractions, with user-oriented representations at the higher (less detailed) levels and a database designer-oriented representation at the detailed bottom level.
- Diagrams using components are easier to comprehend. The user becomes familiar with components, rather than large, difficult-to-follow diagrams, which also facilitates the communication among users and designers.
- The designers are given support for reusing their design knowledge and experience to represent similar situations. This contributes to speeding up the modeling phase.
- In a multidimensional environment, like the spatiotemporal, which is characterized by frequent spatial, temporal, and spatiotemporal aspects, it is often desirable to focus on one aspect at a time. Using patterns facilitates this, as it is possible to concentrate on one aspect–for example, the spatial –temporarily ignoring the temporal aspect, which is captured using temporal patterns.

2.2 Modeling Patterns versus Object-Oriented Design Patterns

The use of the term "pattern" in this paper is inspired by the design patterns that are very popular in object-oriented software development (Pree, 1994) (Gamma, 1992) (Fowler, 1997). Here, we contrast our data modeling patterns with design patterns, pointing out differences and similarities.

Although there exists no standard definition of the term "design pattern" in object-oriented software development (Pree, 1994), it should be clear that both design patterns and our modeling patterns are generic abstractions that have as purpose to reduce the complexity of software development. However, a fundamental difference between the traditional design patterns and ours is that the former concern software development in a broad sense, while our patters concern only the aspect of (conceptual) data modeling, where the aim is limited to capturing the structural aspects of the modeled reality with the purpose of capturing these in a database.

Design patterns include sets of rules describing how to accomplish identified tasks in the realm of software development. The closest parallel to our modeling patterns and their corresponding components is the rules in Section 4.2, Table 2, that describe their use in the conceptual data modeling task.

Design patterns are characterized by their *purpose* and the used *notation*. In the object-oriented software development area the purpose of creating patterns is to produce reusable single components in program codes. For this, the purpose stresses the framework aspect and is related to the programming language used. The notation of a pattern can be (a) informal textual (i.e., using natural language), (b) formal textual (i.e., using a formal language or a programming language) or (c) graphical. In our approach the purpose is similar: we want to produce reusable parts of conceptual schemas. The framework of our modeling patterns is the ER model, i.e., this is the language we chose use, and the notation is both informal textual (see the natural language description of patterns in Section 4) and graphical notation (using the ER model).

Finally, Gamma et al. (Gamma, 1992) have proposed a catalog-like presentation of more than 20 design patterns and view these as a common vocabulary for software design. We view the proposed 45 modeling components (i.e., the abbreviation of modeling patterns) in a similar way: once the designer identifies the spatial, temporal, or spatiotemporal aspect of a construct, it is possible to apply the appropriate pattern, following the rules of Table 2.

Note that, putting aside differences and similarities with the design patterns, the set of proposed modeling patterns and components have the goal of reducing the complexity of the conceptual modeling phase for spatiotemporal applications for the designer, as well as simplifying the resulted conceptual schema, in favor of the user. There is no code accompanying the description of each pattern (as is usual for object-oriented design patterns) as their role is restricted to semantic level, with no connection to implementation details.

3. Requirements to Spatiotemporal Applications

This section formulates spatiotemporal requirements to the ER model. We include them here because they drive the derivation of the patterns. In Section 4, we propose corresponding patterns.

The requirements are based on theoretical studies (Tryfona and Hadzilacos, 1995) (Tryfona and Jensen, 1998) as well as on practical experience from two real-world applications: one concerning the design and development of a network utility management system (Utilnets, 1994), and one dealing with the design of a cadastral database (Cadastral, 1997). The objective underlying the requirements is that patterns should exist to allow the designer to capture meaningful spatiotemporal aspects relevant to the application.

The ER model, like other conventional conceptual data models, encompasses three fundamental modeling constructs, namely entity sets, attributes, and relationship sets. Entity sets are used for modeling objects, which have independent existence; attributes describe properties of objects and relationships; and relationship sets capture relationships between objects. The spatiotemporal aspects we consider are spatial extents (*where* something *is*), existence time (*when* something *exists*), valid time (*when* something is *true*), and transaction time (*when* something is *recorded as current* in the database) (Snodgrass and Ahn, 1985) (Jensen et al., 1998). These aspects include those covered by the vast majority of works.

Table 1 states which spatiotemporal aspects are meaningful to capture for each modeling construct. A "-" indicates that the combination is not meaningful. The letters refer to the discussion that follows.

	Spatial Extent	Temporal Extent			Spatiotemporal Extent
		existence time	valid time	transaction time	
Object	a	b	–	b	c
Attribute	d	–	e	e	f
Relationship	g	–	h	h	i

Table 1: Objects, attributes, and relationships in a spatiotemporal environment.

a. Associations of *objects with space* are meaningful and important in many applications. For example, in a cadastral system, we want to capture the spatial extent, or position, of a "landparcel."

b. Two *temporal extents of objects* are meaningful.

- The *existence time*, describing when an object exists in the real world. For example, a "landparcel" LP may exist from 1890 to 1980.
- The *transaction time*, giving the time an object is recorded as current in the database. For example, although "landparcel" LP started to exist in 1890, it was recorded in the cadastral database as an entity in 1930.

Objects do not have a truth value, as they are not logical statements. And so, valid time is not a meaningful aspect of an object.

c. Note also that these three spatiotemporal aspects of an object may be combined freely. For example, the existence of "landparcel" LP is recorded in time, together with its various positions (in time).

d. It is also meaningful to record the spatial extent of an attribute of an object or relationship (i.e., the spatial extent of the association of an attribute value with an object or a relationship).

Attributes such as "soil type" or "erosion" belong to space itself, rather than to specific objects. We therefore call them *spatial properties* or *spatial attributes*. Spatial attributes indirectly become properties of spatial objects via their position in space, i.e., the spatial objects inherit them from space. For example, although one application may view the "soil type" of "landparcel" LP as an attribute of the landparcel, it is clear that: (a) the "soil type" is defined whether or not the landparcel exists at that position in space, and (b) when the landparcel moves (or changes shape), the landparcel's "soil type" will not remain unchanged; rather the "soil type" attribute inherits new values from the new position. For spatial attributes, we need to describe the minimum unit in space in which the value of the attribute remains constant. For example, "soil type" changes per region, while "elevation" changes per point in space.

This paper focuses on the spatial extents of attributes of space and will not consider spatial extents of other attributes; these appear somewhat artificial. For example, the name of some specific country may vary over space; in German speaking regions, one name is used, while other names may be used in English or Portuguese speaking regions. We do not provide patterns for capturing the variation of such attributes over space. Finally, the discussion for object attributes also applies to relationship attributes.

e. It is meaningful and of fundamental importance to be able to record when an attribute value is associated with an object or a relationship. Again, there are two types of time in which we want to record these properties.

- The *valid time*, showing the time the property (attribute) of the object (or relationship) is true in time. For example, the "use" of "landparcel" LP from 1890 to 1940 was agricultural.

- The *transaction time*, showing the time the property is part of the current state of the database. For example, the "use" of LP was recorded as agricultural in 1930 (when the cadastral database was created).

f. As before, the spatiotemporal aspects may be combined freely. For example, we need to record the changes of spatial attributes (of objects or of space) over time, such as the way the "soil type" of an area changed in the last ten years.

g. In the same way as for attributes, it is also meaningful to record the spatial extent of a relationship. Certain relationships among objects may only be considered true in certain parts of space. The spatial extent of a relationship is not necessarily dependent on the spatial extents of the objects that the relationship is among, or even dependent on the objects having spatial extent. An example of the latter is a "has" relationship between car models (with no spatial extent) and car ratings (with no spatial extent). The spatial extent of relationship "has" between object "Model 1" and object "approved" is that of the European Union.

We will focus on the relationships that are similar to the spatial attributes discussed in Item d above. These *spatial relationships* are determined in terms of the spatial extents of the objects being related. For example, river R "traverses" landparcel LP is a spatial relationship. The spatial relationships are subdivided into three subsets: topological (e.g., "inside," "outside"), directional (e.g., "North of," "North-East of") and metric (e.g., "5 km away from") relationships. A conceptual design model should provide built-in support for representing such spatial relationships.

h. As for attributes, it is meaningful and important to be able to associate temporal aspects with relationships. For example, the relationship between "landparcel" LP and its "owner" may have times associated with it. Again, there are two types of time in which we want to record these properties.

- The *valid time*, showing the time the relationship is true in the real world. For example, Jim is the owner of "landparcel" LP since 1900.
- The *transaction time*, showing the time the relationship is part of the current state of the database. For example, the information that Jim is the owner of "landparcel" LP since 1900, was recorded in 1940.

i. Again, these spatiotemporal aspects of relationships may be combined freely. Consider the example of two "neighboring landparcels" in time.

Note that spatial attributes and relationships are determined by the objects' spatial extents. The similar temporal (existence time and transaction time) attributes and relationships of objects do not appear to play a significant role in spatiotemporal applications. Finally, it should be observed that the spatial extent of an object represents a property of the object, just like an attribute value does, and thus has a valid time and a transaction time, just like an attribute value does.

4. Modeling Patterns at the Conceptual Level

This section offers conceptual modeling patterns and corresponding components that address the requirements from the previous section. In Section 4.1, for each requirement (a to i) described in Section 3, a corresponding pattern is given, and the ER modeling constructs that make up each pattern is explained. So, by combining objects (or, entity sets), attributes and relationships (or, relationship sets) with spatial, temporal, and spatiotemporal extents, spatial, temporal, and spatiotemporal entity sets, attributes, and relationship sets, respectively[2], result. Then, we represent

[2] For the rest of the paper, for reasons of simplicity and readability, terms "object", "entity set" and "entity" are used interchangeably.

the component, which can be used at a less detailed level, to capture this pattern. Section 4.2 discusses how to use patterns and components to capture spatiotemporal aspects.

4.1 Spatiotemporal Modeling Patterns and Components

For each discussed pattern, we present the corresponding component. A component that has built-in support only for a spatial aspect, is termed *spatial*; if it captures a temporal aspect is called *temporal*; and if it captures both, it is *spatiotemporal*. The lower right corner of the component, in the circle, indicates the spatial dimension, the upper right the temporal one, and both, the spatiotemporal.

a. Spatial Entity Set

Modeling Pattern.

Objects in spatiotemporal applications have positions in space (i.e., spatial objects), and it is frequently necessary to capture this position in the database. The first step to support this is to provide means for representing the space in which the objects are embedded. The next is to provide means for indicating that the objects' positions in this space are to be captured. For these purposes, we use the following special entity and relationship sets.

(i) The special entity sets SPACE, GEOMETRY, POINT (or "P"), LINE (or "L"), REGION (or "R"). Entity set GEOMETRY captures the geometrical position of the entity set and can be POINT, LINE, REGION, or any other geometric type (or geometry). For simplicity we use only POINT, LINE, REGION, and their combinations. POINT, LINE and REGION may occur simultaneously and represent then different views of the same entity set.

(ii) The special relationship set "is_located_at" that associates a spatial entity set with its geometry. The cardinality of this set is 1:M because a spatial entity may have more than one geometry when multiple granularities are employed. The relationship set "belongs_to" between GEOMETRY and SPACE with cardinality constraint M:1 is also included.

Figure 2 illustrates the pattern representing objects in space.

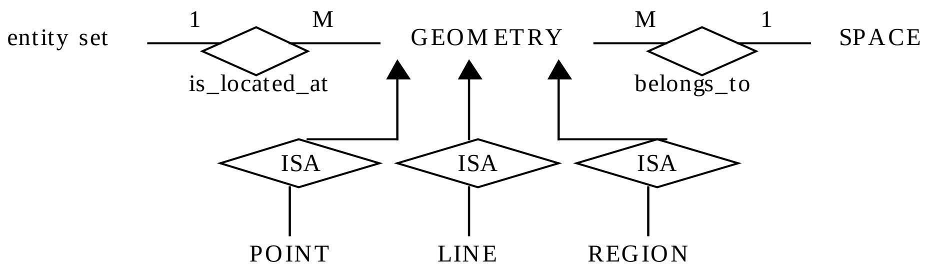


Figure 2: A conceptual modeling pattern representing objects in space.

Component.

The corresponding component is illustrated in Figure 3(a), while 3(b) shows a "landparcel" as REGION and/or as POINT in space. To capture the spatial aspect of the entity set, a circle with an "s" is used in the lower right corner; when this becomes specific, like in the case of "landparcel," then it shows the real geometry (e.g., "P" or "R").

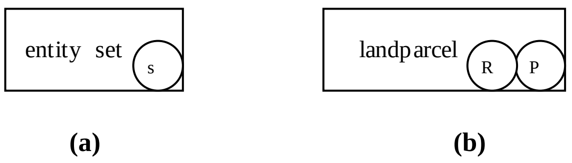


Figure 3: (a) a component showing spatial entity sets, (b) a landparcel as REGION and/or POINT.

b. Temporal Entity Set

Modeling Pattern.

Entity sets can be assigned existence and transaction time. Since the existence time of an entity captures the time the corresponding real-world object exists in the miniworld, the existence time must be associated with the "id" of the entity, because the "id" identifies existence. Attributes connected to each other denote composite attributes (Elmasri and Navathe, 1994). Thus (Figure 4(a)) "id/time" values consist of pairs of "id" values and "existence time" (or "et") where “existence time” is a pair of “start” and “end” values. Transaction time captures the time during which the entity is current in the database. The transaction time of the entity must, as before, be associated with the "id" of the entity. That is, this case is modeled by using the same pattern as before where "existence time" is replaced by "transaction time" (or "tt"). If there is a need to record both the existence and transaction time for an entity the "id/time" values consist of "existence time", “transaction time”, and "id" values (Figure 4(b)).

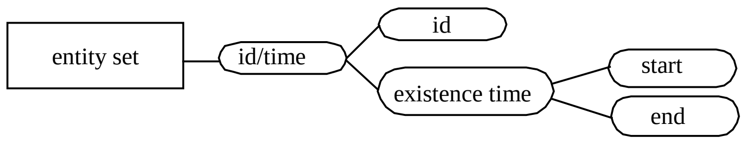


Figure 4(a): a conceptual modeling pattern representing objects’ existence in time.

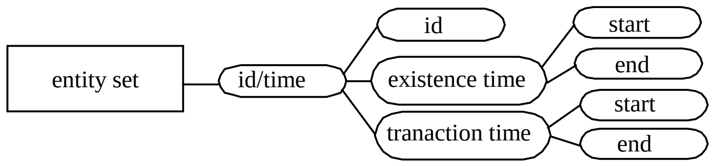
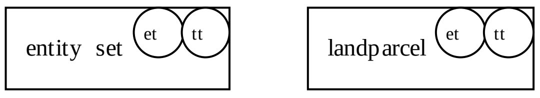


Figure 4(b): a conceptual modeling pattern for capturing objects’ existence and transaction time.

Component.

In the corresponding component, the temporal dimension is captured with an "et", or "tt" or both in the upper right corner of the entity set. The corresponding component of Figure 4(b) is illustrated in Figure 5(a), while 5(b) shows a temporal "landparcel."



(a) **(b)**

Figure 5: (a) a component showing temporal entity sets, (b) landparcels as temporal entity sets.

c. Spatiotemporal Entity Sets

There are two cases one should distinguish here.

Pattern (i).

Spatial objects with temporal support which refers to their existence and recording (i.e., transaction) in time. In this case, "id/time" is connected to the entity set, as previously. The detailed ER is depicted in Figure 6.

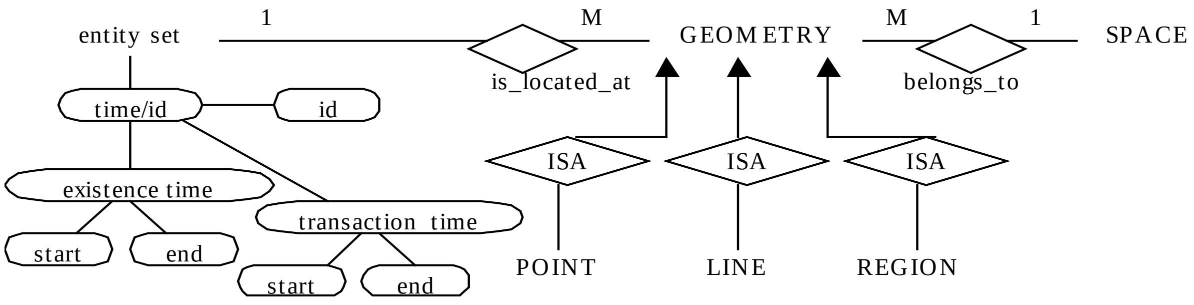


Figure 6: A conceptual modeling pattern for capturing spatial objects' existence and transaction time**.**

Component (i).

Figure 7(a) gives the corresponding component and Figure 7(b) an example.

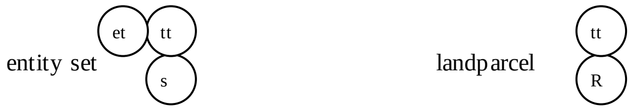


Figure 7: (a) a component showing entity set with temporal support, (b) spatial lanndparcels and their transaction time.

Pattern (ii).

Spatial objects with temporal support which refers to their position in time. When an object changes its position over time, it is the geometry that changes rather than the object itself, that is, the relationship between the object and its geometry. To capture a temporal aspect of the positions of the objects in an entity set, the entity set itself is not annotated but the relationtionship "is_located_at" is annotated. The consequence of this is that existence time is not used to capture the change since it is not possible to associate existence time with relationship instances. Therefore, a "valid time" (or "vt"), or "transaction time" (or "tt") or both attributes are connected to the relationship "is_located_at". The first notation ("valid time") indicates valid-time support: the objects' current positions as well as their past and future positions are to be captured. The second annotation ("transaction time") indicates transaction time support: the current positions as well as all positions previously recorded as current in the database are to be captured. Figure 8 illustrates both valid time and transaction for spatial objects.

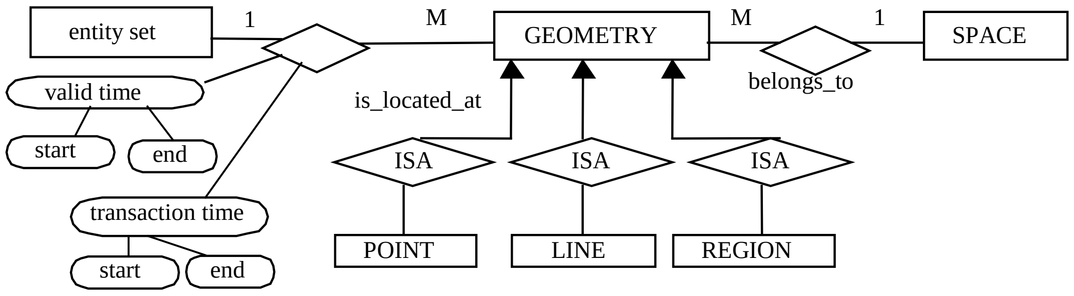


Figure 8: A conceptual modeling pattern representing objects with temporal support of their position.

Component (ii).

The corresponding component of Figure 8 is illustrated in Figure 9(a), while 9(b) shows a "waterpipe" and its recording position in time.

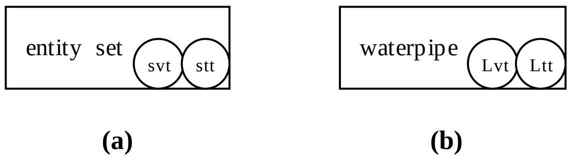


Figure 9: (a) a component showing spatial entity set with temporal support of its position, (b) recording waterpipe's position in time.

Finally, when the combination of (i) and (ii) exists, it shows spatial objects with temporal support for their existence and transaction in time as well as temporal support for their positions.

d. Spatial Attribute

Pattern.

Attributes may also have associated locations in space (*spatial attributes*), which are described as sets of geometric figures (Section 3.1). In terms of the ER model, a spatial attribute is modeled as an entity set with a composite attribute "attribute/spatial_unit", consisting of the "attribute value", and the "spatial unit" which represents the geometry in which the "attribute value" is constant. So, the spatial unit can be "POINT" (or "P), "LINE" (or "L") and "REGION" (or "R") geometric type. The spatial attribute is further connected to SPACE via the relationship set "has_spatial_attribute." In this way, each part of space is assigned a specific value of the attribute. By connecting a spatial entity set to GEOMETRY (via the special relationship "is_located_at," see previous figures) and GEOMETRY to SPACE (via "belongs_to"), an object inherits spatial attributes. In other words, spatial attributes of entities are *derived properties* from space (indicated as shaded). Figure 10 depicts the modeling pattern.

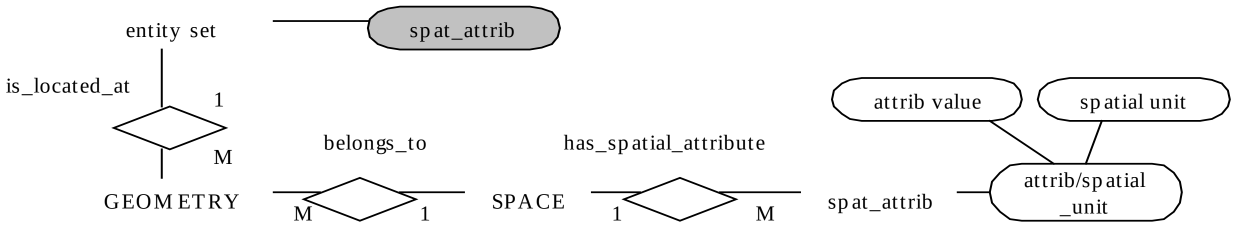


Figure 10: A conceptual modeling pattern to represent spatial attributes.

Component.

The corresponding component is illustrated in Figure 11(a), while 11(b) shows that the "soil type" value of a landparcel is associated with a set of spatial regions.

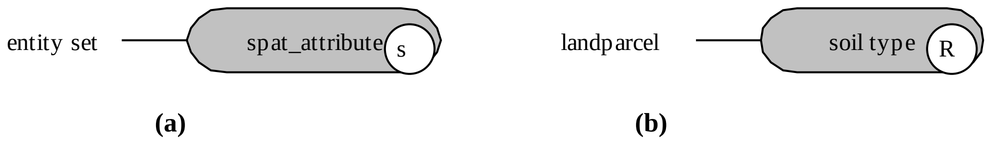


Figure 11: (a) a component showing a spatial attribute, (b) "soil type" as a spatial attribute.

e. Temporal Attribute

Pattern.

Values of attributes of entities denote facts about the entities and thus have both valid and transaction time aspects. The captured valid and transaction time of an attribute has to be associated directly to the attribute value. Thus a temporal attribute is modeled, in terms of the ER model, as a composite attribute consisting of two components; the attribute capturing the atrribute values and the time capturing the needed temporal aspects of that specific attribute value. Figure 12 illustrates the modeling pattern for this purpose.

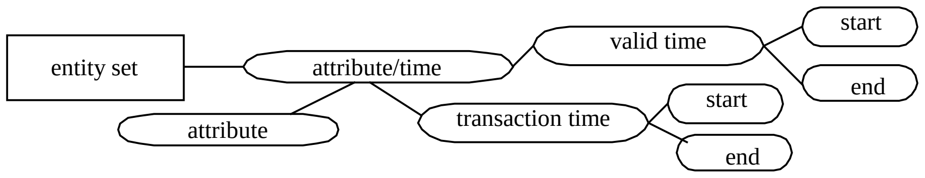


Figure 12: A conceptual modeling pattern to represent attributes in time.

Component.

The corresponding component is illustrated in Figure 13(a), while 13(b) shows "color" as temporal attribute.

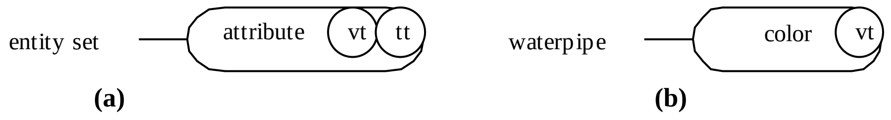


Figure 13: (a) a component showing a temporal attribute, (b) "color" as a temporal attribute.

f. Spatiotemporal Attribute

Again, here, like in case (c), two cases should be distinguished.

Pattern (i).

Spatial attributes with temporal support which refers to attributes' valid and transaction periods in time (i.e., the spatial attribute is treated as a normal attribute in time). The detailed ER is depicted in Figure 14.

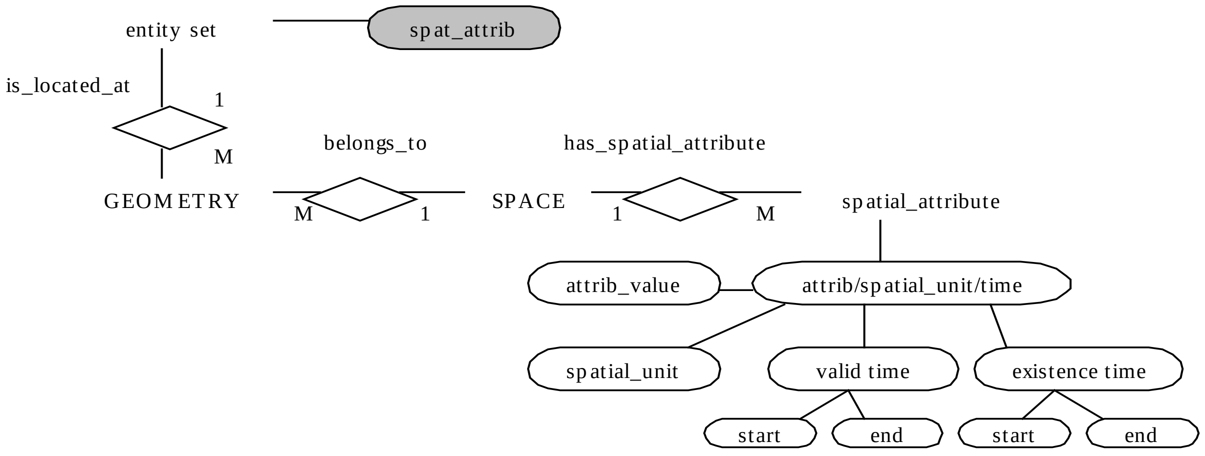


Figure 14: A conceptual modeling pattern spatial attributes in time.

Component (i)

Figure 15(a) gives the corresponding component and Figure 15(b) an example.

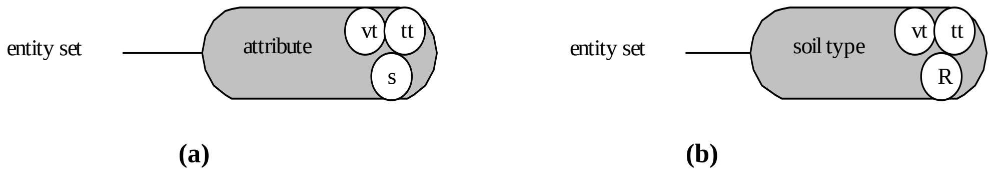


Figure 15: (a) a component showing a spatial attribute with temporal support, (b) "soil type" as a spatial attribute with temporal support.

Pattern (ii).
Spatial attributes with temporal support which refers to attributes position valid and transaction periods in time. In this case, the temporal aspects (valid and transaction time) of spatial attributes are recorded by placing "valid time," or "transaction time", or both in the relationship "has_spatial_attribute" (Figure 16).

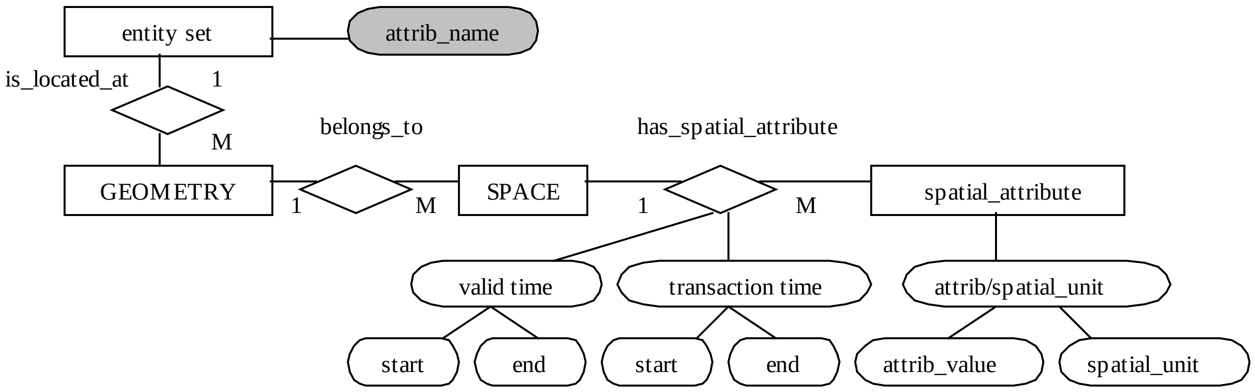


Figure 16: A conceptual modeling pattern representing spatial attributes with valid time support.

Component (ii).

The corresponding component is illustrated in Figure 17(a), while 17(b) shows "soil type" as a spatiotemporal attribute with valid time support.

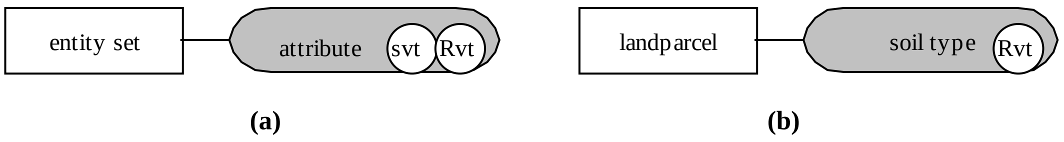


Figure 17: (a) a component showing spatial attribute with temporal support for its spatial part, (b) "soil type" as a spatiotemporal attribute.

Finally, the combination of (i) and (ii) cases captures spatial attributes with temporal support for the attributes themselves, as well as their spatial dimension.

Attributes can be associated to entity sets, as well as to relationship sets. Here, we described only attributes associated to entity sets; the ones connected to relationship sets are modeled in a similar way.

g. Spatial Relationship Set

Pattern.
Spatial relationships are special kinds of relationships. In particular, they are associations among the geometries of the spatial entities which, for reasons of simplicity and ease of understanding, are described as relationships among the spatial entity sets themselves. Figure 18 shows the way geometries are related under spatial relationships.

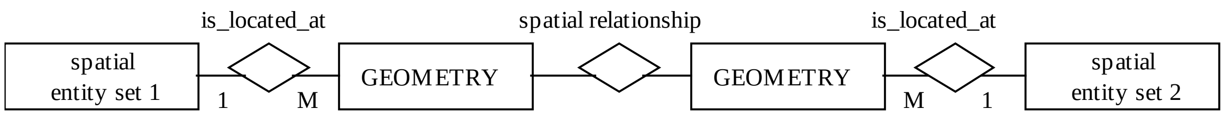


Figure 18: A conceptual modeling pattern to represent spatial relationships.

Component.

Figure 19(a) shows the design pattern of the general case and 19 (b) an example of, the relationship "traverses" between cities and rivers relating the geometries of entities of these two spatial entity types.

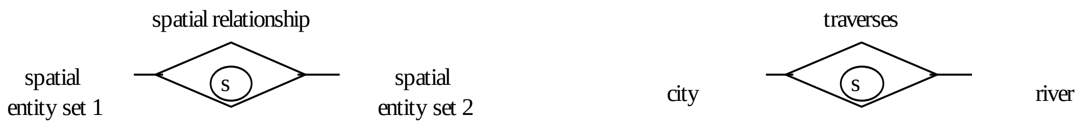


Figure 19: (a) a component showing spatial relationships, (b) "traverses" as a spatial relationship.

h. Temporal Relationship Set

Pattern.

By annotating a relationship with a temporal aspect (valid time, transaction time, or both), we capture the changes of that temporal aspect for the set's relationships. Figure 20 depicts a temporal relationship.

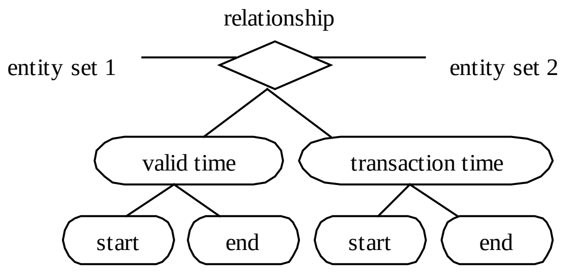


Figure 20: A conceptual modeling pattern represent to represent temporal relationships.

Component.

Figure 21 shows the corresponding component and an example.

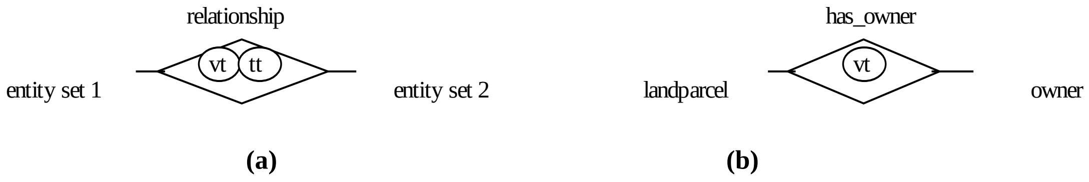


Figure 21: (a) a component showing temporal relationships, (b) "ownership" as a temporal relationship.

i. Spatiotemporal Relationship set

Pattern.

A spatiotemporal relationship is a spatial relationship with time support. In particular, by annotating a spatial relationship with a temporal aspect we capture the changes of the spatial relationship over time. Figure 22 shows the general representation of a spatiotemporal relationship.

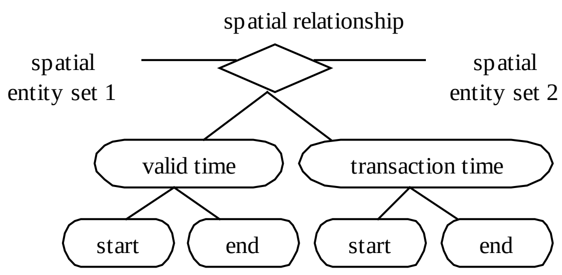


Figure 22: A conceptual modeling pattern to represent spatiotemporal relationships.

Component.

Figure 23 depicts changes of the relationship "traverses" between cities and rivers are recorded in time.

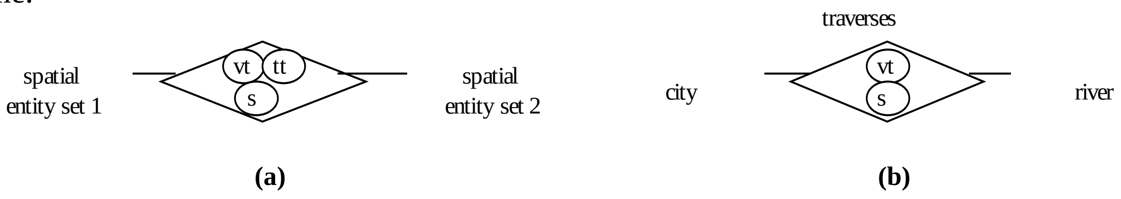


Figure 23: (a) a component showing spatiotemporal relationships, (b) "traverses" as a spatiotemporal relationship (valid time support).

In the next section we discuss how patterns and components can be used by the designer, and how they operate in a tool supporting ER functionality.

4.2 A Library of Modeling Components and Rules of Usage

The main objective behind the approach of recognizing patterns and replacing them by components is to aid the designer capture the appropriate spatiotemporal information in a detailed diagram, and help the user understand semantically rich diagrams. The outcome of using components is diagrams with reduced complexity. Modeling patterns and components can be used in two ways:

- *bottom-up*: the designer captures all the information of a specific application in a detailed diagram and then, by marking (i.e., recognizing) the appropriate constructs (i.e., forming a pattern, as shown in Section 4.1), replaces them by the corresponding component;
- *top-down*: the designer and/or the user adds the spatial, temporal, or spatiotemporal aspects to a schema by plugging in the appropriate components.

In order to support patterns and components, an ER design tool should be accompanied by the library of components. In this way, both the designer and the user can follow the bottom-up or top-down approach to capturing their application. Each row in Table 2 represents the component and the number of possible combinations based on the number of symbols each component contains. For example, for the temporal entity set with "et" and "tt", either one (e.g., "et") or the other (e.g., "tt") or both can be present. When both dimensions are present then symbols from both have to participate in the component. For example, for the spatiotemporal entity set with "et", "tt", and "s" the result is, either "et" and "s"; "tt" and "s"; or "et", "tt", and "s". The total number of components is 45.

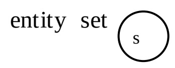
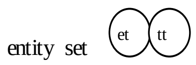
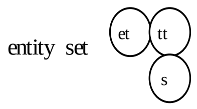
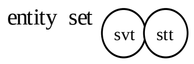
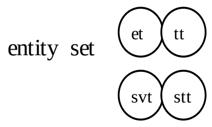
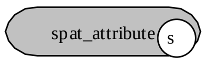
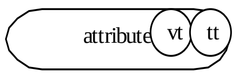
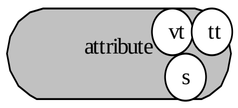
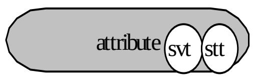
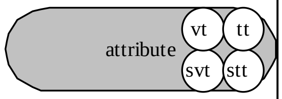
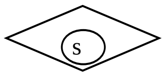
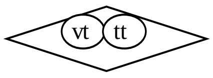
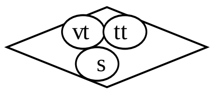
	Component	# of combinations	Rule of Usage
Spatial Entity Set		1	Entity set with spatial extent.
Temporal entity Set		3	Entity set for which existence time and transaction time are recorded.
Spatio Temporal Entity Set		3	Entity set, for which existence time and transaction time are recorded, with spatial extent
Spatio Temporal Entity set		3	Entity set, with spatial extent for which valid time and transaction time are recorded.
Spatio-temporal Entity set		9	Entity set, with spatial extent for which valid time and transaction time are recorded. Existence time and transaction time of the entity set are also recorded.
Spatial Attribute		1	Attribute with spatial extent.
Temporal Attribute		3	Attribute for which valid time and transaction time are recorded.
Spatio Temporal Attribute		3	Attribute for which valid time and transaction time are recorded. The attribute has also spatial extent.
Spatio Temporal Attribute		3	Attribute, with spatial extent for which valid time and transaction time are recorded.
Spatio Temporal Attribute		9	Attribute, with spatial extent for which valid time and transaction time are recorded. Valid time and transaction time of the attribute are also recorded.
Spatial Relationship Set		1	Relationship set with spatial extent.
Temporal Relationship Set		3	Relationship set for which valid time and transaction time are recorded.
Spatio Temporal Relationship Set		3	Relationship set with spatial extent. Valid time and transaction time of the relationship set are also recorded.

Table 2: A library of spatial, temporal and spatiotemporal components, and the rules of their usage.

5. Example of Use

In this section we give, an example from a real cadastral system (Cadastral, 1997). First, we model (Figure 24) the application requirements at the conceptual level by using the proposed spatial, temporal, and spatiotemporal components and not using them (Figure 25).

"Parts of land (i.e., landparcels) or rivers are typical cadastral units. For each unit the system keeps track of the time of existence as well as its time of recording in database. Landparcels can have agricultural use, or be buildings. Buildings have owners, who may change in time. If a landparcel has an agricultural use, then vegetation and soil type is recorded in the database and in real world respectively, together with their geometries in space. For landparcels, soil type, vegetation, and elevation are recorded; the first two in terms of regions, while the last one in terms of points. Landparcels can be crossed by rivers at different periods in time. Finally, landparcels ownerships are recorded in time; for owners, SSN and name is known."

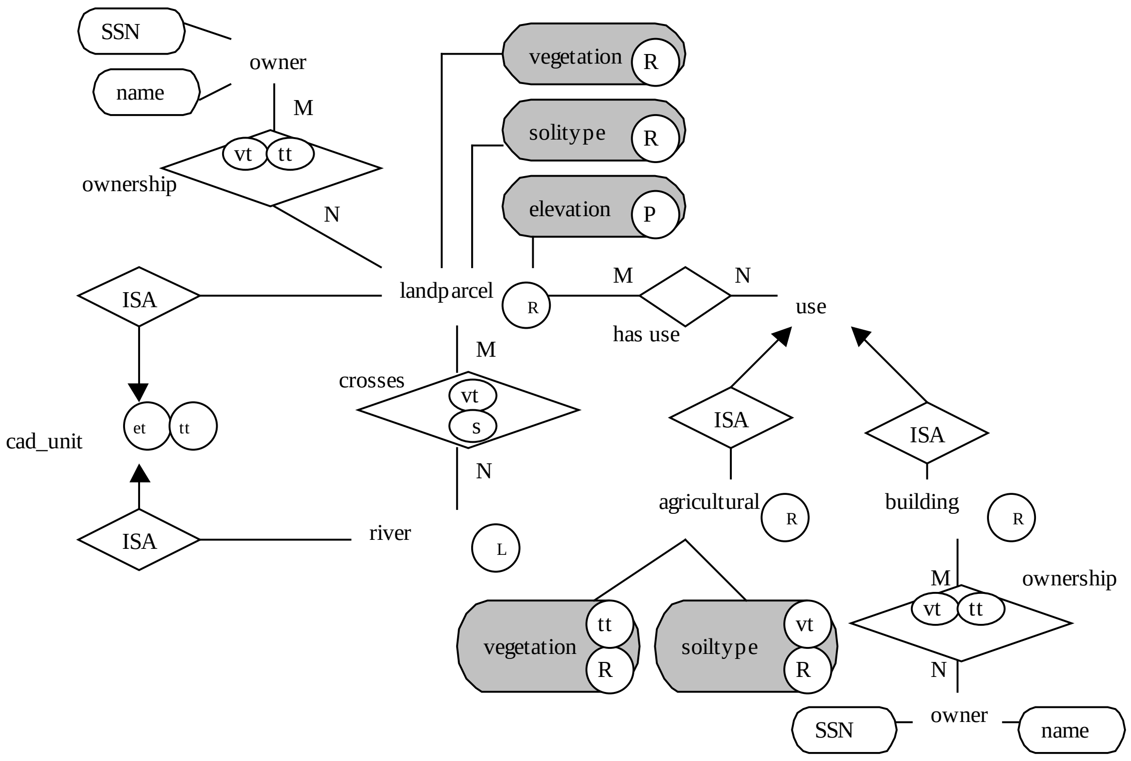


Figure 24: A conceptual schema of a cadastral example, by using components.

The representation of Figure 24 is easy for the user to follow, free of details, and less complex, but it still captures the intended semantics. Also, for the designer, it is easier to capture the spatiotemporal aspects by plugging in the appropriate component. For example, since elevation is recorded in points, there is a spatial attribute with the ("P") indicator in the lower right corner. On the contrary, Figure 25, which describes all the above information in the detailed ER, is difficult to follow.

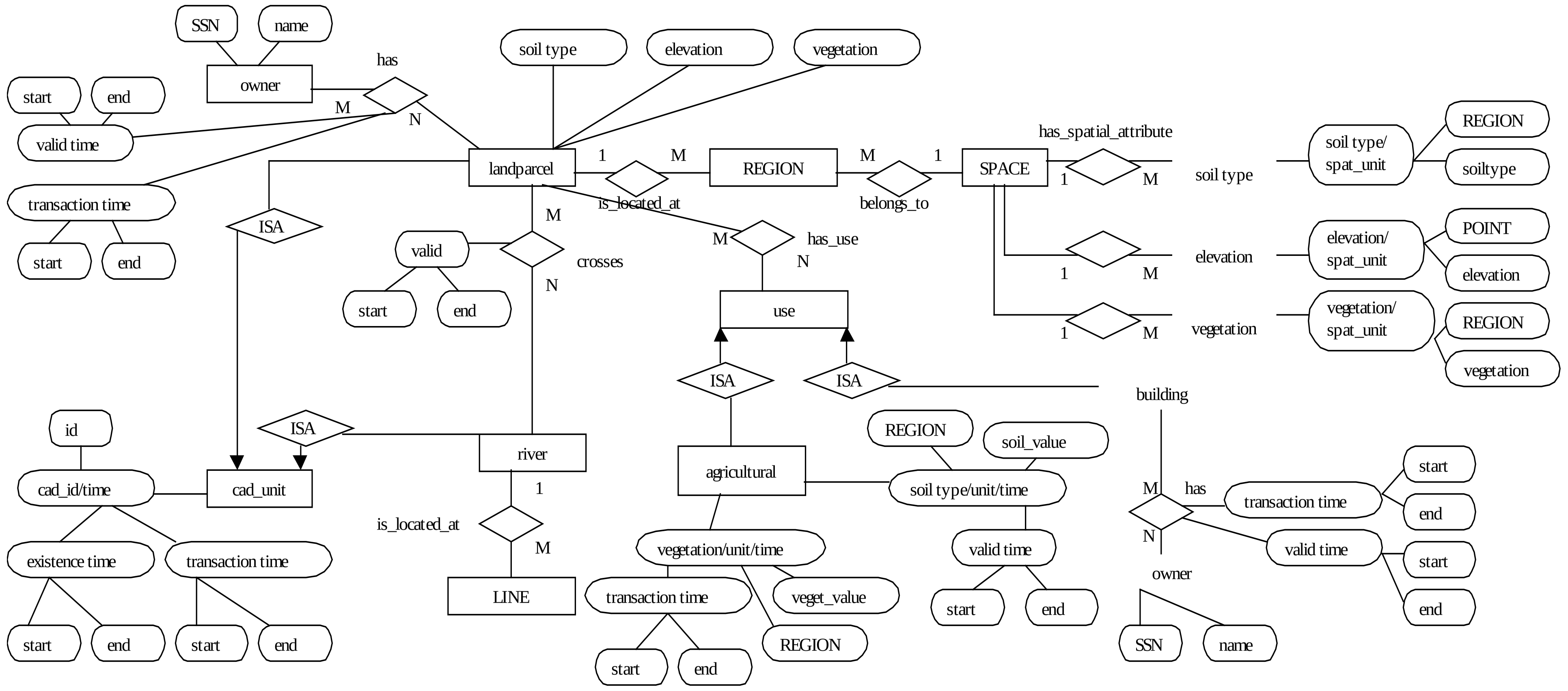


Figure 25: The detailed representation of the cadastral example in ER.

6. Conclusions and Research Directions

This paper considers the use of modeling patterns (i.e., autonomous, semantically meaningful conceptual diagrammatic parts) and their corresponding components for conceptual design for spatiotemporal applications. Based on a well-defined set of spatiotemporal requirements, the paper offers a set of modeling patterns that appear multiple times in conceptual diagram, capturing the spatial, temporal, and spatiotemporal aspects relevant to of the particular application. We propose the replacement of them by corresponding components. The result is more elegant diagrams that are easy to follow, less complicated, but yet semantically rich. As a prototypical model for this approach, we use the Entity-Relationship, and the paper includes a library of spatiotemporal components. An example from a real cadastral system shows the benefits of the component-based approach.

We are currently implementing (Andersen et al., 1998) a design tool that supports the approach, in which the users can draw diagrams using components, and then, semi-automatically, translate the component-based diagram into a logical schema.

This research may be extended in several directions. We are considering rules for combining components at the same level of abstraction: for example, what are the governing rules in connecting a spatial entity set to a spatiotemporal entity set component? What spatiotemporal semantics are involved? Furthermore, how can one combine patterns (or components) to produce complex patterns (or complex components), to further facilitate the design process for complicated spatiotemporal applications. This also depends on the represented semantics. Finally, another important issue is how large a complex pattern can be, and how abstract, but still semantically rich, components should be provided.

Acknowledgements
The authors thank Dr. M. Kavouras and the National Technical University of Athens, Greece, for providing the environment for the Cadastral application case study. Also, discussions on temporal issues with H. Gregersen were significantly helpful.

7. References

1. Abiteboul, S., and Hull, R., 1987. *IFO: A Formal Semantic Database Model*. ACM TODS, 12(4):525-565.
2. Andersen, S, Mogensen, S., and Tryfona, N., 1998. *A Case tool for Spatiotemporal Application Design.* Aalborg University Report,.
3. Cadastral, 1997. *Definition of a Standard for the Exchange of Digital Cadastral Data*. National Technical University of Athens funded by CADASTRE SA, Greece.
4. Castano, S., De Antonellis, V., and Zonta, B., 1992. Classifying and Reusing Conceptual Schemas. In G. Pernul and A. A. Tjoa, editors, ER'92 - Entity-Relationship Approach, Springer-Verlag 645.
5. Castano, S., and De Antonellis, V., 1994. Standard-Drive Re-engineering of Entity-Relationship Schemas. In P. Loucopoulos, editor, ER'94 - Buisiness Modelling and Re-Engineering, Springer-Verlag 881.
6. Chen, P.S., 1976. *The Entity-Relationship Model: Toward a unified view of Data*. ACM TODS, 1(1):9-36.
7. Claramunt, C., Parent, C., and Theriault, M., 1997. *Design Patterns for Spatiotemporal Processes*. IFIP 1997. Chapman and Hall.
8. Delcambre, L., and Langston, J., 1996. Reusing (Shrink Wrap) Schemas by Modifying Concept Schemas. In Stanley Y. W. Su, editor, Proc. of the 12th International Conference on Data Engineering (ICDE'96), IEEE Computer Society.

9. Fowler, M., 1997. Analysis Patterns: *Reusable Object Models*. Addison-Welsey.
10. Gamma, E., Helm, R., Johnson, R., and Vlissides, J., 1994. *Design Patterns*. Addison-Welsey.
11. Gordillo, S., Balaguer, F., and Das Neves, F., 1997. *Generating the Architecture of GIS Applications with Design Patterns*. Proc. of the 5th ACM International Symposium on Advances in Geographic Information Systems. Las Vegas, Nevada, Nov. 1997.
12. Jensen, C. S., and Dyreson, C. E., (editors) 1998. *The Consensus Glossary of Temporal Database Concepts*. In Opher Etzion et al. (editors): Temporal Databases: Research and Practice. LNCS: 367-405, Vol. 1399, Springer 1998.
13. Hay, D. C., 1996. Data Model Patterns. Dorset House.
14. Hull, R., and King, R., 1987. *Semantic Data Modeling: Survey, Applications and Research Issues*. ACM Computing Survey, 19, 3.
15. Pree, W., 1994. *Design patterns for Object-Oriented Software Development*. Addison-Wesley.
16. Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W. 1991. *Object-Oriented Modeling and Design*. Prentice-Hall: Englewood Clifs, NJ.
17. Snodgrass, R. T., and Ahn, I., 1985. *A Taxonomy of Time in Databases*. SIGMOD Conference 1985: 236-246
18. Story, P.A., and Worboys, M.F., 1995. *A Design Support Environment for Spatio-Temporal Database Applications*. Proc. of the International Conference on Spatial Information Theory (COSIT'95), Lecture Notes in Computer Science (988):413-430.
19. Teory, T.J., Guangping, W., Bolton, D., and Koenig, J., 1989. *ER Model Clustering as an Aid for User Communication and Documentation in Database Design*. Communications of ACM, 32(8).
20. Tryfona, N., and Hadzilacos, Th., 1995. *Geographic Applications Development: Models and Tools at the Conceptual Level.* Proc. of the 3rd ACM International Symposium on Advances in Geographic Information Systems. Baltimore, Maryland, Dec. 1995.
21. Tryfona, N., and Jensen, C. S., 1998. *Conceptual Data Modeling for Spatio-Temporal Applications*. Chorochronos Technical Report CH-98-8. Under submission.
22. UtilNets, [1994-1997], *completed*. Work Program of Brite-Euram Project 7120, DG XII, European Union, Brussels, Belgium.