Object-Based Modeling and Language for an Object-Oriented Spatio-Temporal Database System

Yang Hee Kim[†]

ABSTRACT

In this paper, we present an object-based modeling and language for an object-oriented spatio-temporal database system. For handling the structure of spatio-temporal objects and the spatio-temporal operators, we propose the two layers of data modeling: a *spatio_temporal object model* (STOM) and an *spatio_temporal internal description model* (STIM). We then propose STOQL, a spatio-temporal object-oriented query language. STOQL provides an integrated mechanism for the graphical display of spatial objects and the retrieval of spatio-temporal and aspatial objects.

Keywords : object-oriented spatio-temporal databases, object-based approach, object-oriented data modeling, spatio-temporal object-oriented query language

객체지향 시공간 데이터베이스 시스템의 객체기반 설계 및 질의어

김양희[†]

요 약

본 논문에서는 객체지향 시공간 데이터베이스 시스템의 데이터 모델링과 질의어를 객체지향 기법을 사용하여 소개한다. 시공간 객체와 시공간 연산자를 다루기 위해 다음과 같은 두 단계 객체지향 데이터 모델을 제안 한다: 시공간 객체 모델과 시공간 내부 기술 모델. 또한 객체지향 시공간 질의어인 STOQL을 제안한다. STOQL은 공간 객체의 다양한 출력과 시공간 및 비 공간 객체의 검색을 수행할 수 있는 통합 기능을 제공해준다.

키워드 : 객체지향 시공간 데이터베이스, 객체지향 기법, 객체지향 데이터 모델링, 객체지향 시공간 질의어

1. Introduction

There is a various applications that manipulate objects that change their spatial features through time. Therefore, the change of information becomes important information itself. In particular, information about the position and extent of objects is interested in many different areas of

† 정 회 원 : 한국체육대학교 교양과정부 부교수(교신저자)
논문접수: 2006년 8월 10일, 심사완료: 2007년 2월 12일
이 논문은 2005년도 한국체육대학교 자체 학술연구비의 지원에 의하여 연구되었음

sciences and elsewhere. Examples are GIS applications involving time, and real time applications for vehicle tracking, weather prediction, etc. In this sense, a spatio-temporal databases is a database system that find the changes that objects and their relationships. Furthermore, most of applications require not only the detection of one single changes, but also keep track of the history of changes describing particular evolutions. So spatio-temporal models and languages are capable of handling continuously changing moving objects[4].

Entity-Relationship is the earliest and best known among conceptual database models and has been extended to capture spatio-temporal informations.

The Geo-ER Model[12] is based on the standard ER and the Extended ER models which integrate the concepts of aggregation and grouping. Geo-ER includes special entity sets and relationships to express the semantics of space, the position and space of geographic entities depending attributes and spatial relationships. They use two new constructs called spatial aggregation and spatial grouping. But this model does not satisfy the demanding requirements of spatio-temporal informations.

The Spatio-Temporal Entity-Relationship (STER) model was developed in [17]. STER focuses on modeling units for spatio-temporal applications. Modeling units are part of the conceptual schema, appear repeatedly in the schema, and describe semantically similar aspects. It proposes to replace the units with abstractions, resulting in less detailed, but equally semantically rich, conceptual schemas. However, it lacks the ability to capture the actual motion of the process of change.

The Modeling Application Data with Spatio-temporal features(MADS)[16] is based on object relationship(OR) approach. The implementation of object-relationship models describe processes, which act on the geometric attributes of an entity. This spatio-temporal database processes allows users to handle the complex data model required by more abstracted spatio-temporal applications. Although it is obvious that there is need to represent processes and changes on the conceptual level, there are no exact definitions and operations given. So, the modeling of the processes provides an abstraction on the world, and a description suitable for on application may be inadequate when applied to other applications.

In [8], the constraint-based approach to modeling spatio-temporal data shows that it allows a uniform representation of all kinds of informations. The basic idea of the constraint data model is to represent temporal and spatial objects as infinite collections of points satisfying first-order formula. An embedding of constraints into a relations is performed and also a SQL-like constraint query language is proposed. But constraint query language does not support diverse expression.

Previous works lack uniformity for the representation of spatio-temporal data and an integrated mechanism for the display of spatial objects. To overcome these limitations the object-oriented(OO) technology are being used. Object-oriented database systems provide the incorporation of object-classes that provide a collection of specialized operations. But, it is still necessary to extend object-oriented approach for handling spatio-temporal objects. Especially, spatio -temporal object types must be added to its basic type hierarchy and spatio-temporal operators should be defined over such types. An extension of query language is also required.

In [13], the author proposed three levels of object-oriented data model for handling the structure of spatial objects. She also proposed a spatial object-oriented query language. But this model does not cover temporal informations. It is desirable to have a model which covers not only spatial data but also temporal data.

In this paper, we propose two layers of data modeling : a *physical layer* and a *conceptual layer* for spatial and temporal data. A physical layer is a set of logically related features which are all based on the same topological data structure and a conceptual layer is a set of features that belong to the same theme. In order to support three different types of map layers and sliced representation of temporal objects, we propose the two levels of object-oriented data models: a *spatio_temporal object model (STOM)* for the conceptual layer and a *spatio_temporal internal description model (STIM)* for the physical layer.

We use the OODB ObjectStore[14] and we add several mechanisms for handling spatio-temporal objects to such platform. The STOM gives a formal framework to support spatial and temporal object types and their polymorphic operations which make used of a spatio-temporal query language suitable for manipulation of thematic maps and temporal objects. The notion of generalization is supported in the spatio-temporal object model. The STIM supports efficient and appropriate framework to store geographic coordinates and topology between two geographic objects. And also supports sliced representation to decompose the temporal *evolution* which keep track of the history of temporal changes. The internal description model for spatial object is based on the formal data structure for single-valued vector map and multi-valued vector map[5]. Each physical layer consists of modules and maintains topological integrity. The layer corresponds to one of three topological description levels: geometry level, network topology level, and full topology level, that may match the user's need. Each module consists of internal primitive objects which are the lowest internal object. The generic type of the type hierarchy for the internal primitive objects is vector_type. And vector_type is partition into four subtypes: node, edge, face, and ring.

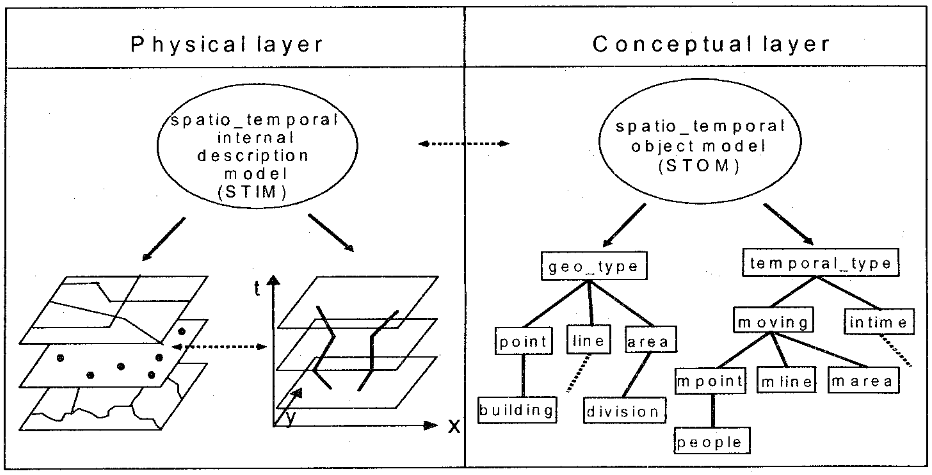
For providing an integrated mechanism for the graphical display of spatial objects and the retrieval of objects, we design a *spatio-temporal object-oriented query language (STOQL).* STOQL is an OQL-based query language and for OQL (Object Query Language), we refer to [1]. To manipulate the presentation of spatial objects, variation of graphical presentations by colors, symbols, and filling are added in STOQL with pull down menus.

This paper is organized as follows. In Section 2 we introduce a two levels of Data Modeling. In Section 3 we present the conceptual layer and physical layer are represented in Section 4. In Section 5 we present the syntax and semantics of the spatio-temporal object-oriented query language, STOQL. The conclusions are presented in Section 6.

2. Spatio-Temporal Data Modeling

A two-layered spatio-temporal data model is used to integrate spatial data, temporal data and aspatial data. Since the application domain of a spatio-temporal data is so wide, we consider modeling in GIS applications involving time.

A temporal geographic information system is a spatio-temporal information system that is designed to work with data referenced by spatial or geographic coordinates upon time[15]. In other words, a temporal GIS is both a database system with specific capabilities for spatio-temporally referenced data, as well a set of operations for working with the data.



<Fig. 1> Two layers of data model

In such a system, the database describes a collection of temporal geographic objects over a two dimensional map with their temporal state. Each temporal geographic object can be classified as belonging to a particular class such as city, road, lake, etc. with their temporal state. That means, we consider them as three dimensional map. Most temporal geographical database systems use multiple map layers to organize geographic objects and sliced representation for temporal objects. There exist two different types of layers: a physical layer and a conceptual layer.

A physical layer is a set of logically related features which are all based on the same topological data structure and a conceptual layer is a set of features that belong to the same theme.

In the following sections, we propose the two layers of spatio-temporal data models: a *spatio_temporal object model* (STOM) for the conceptual layer and an *spatio_temporal internal description model* (STIM) for the physical layer. <Fig. 1> shows the structure of the two layers of data modeling.

3. Conceptual layer

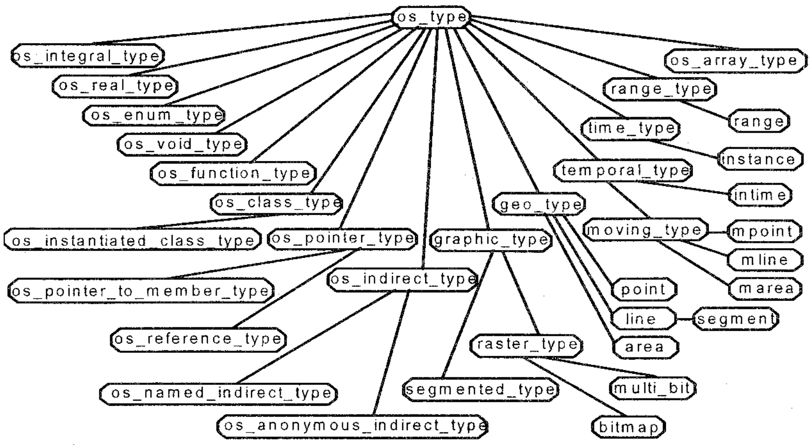
In this subsection, we present the conceptual layer. Conceptual layer is represented by the spatio-temporal object model called STOM. STOM describe a formal framework for modeling the structure of spatio-temporal objects in space and time as well as their relationships, properties, and operations. STOM data model is a collection of spatio-temporal object types(spatio-temporal sorts) and spatio-temporal polymorphic operators. It is based on a signature and an extended signature.

A signature is a pair (S, Σ) with S is a set (whose elements are called *sorts*) and $\Sigma = \{\Sigma_{w,s}\}_{w \in S^*, s \in S}$, is a family of sets (whose elements are called *operators*) and an extended signature is to be introduced for list sorts, product sorts, and function sorts[9].

3.1 Spatio-temporal Object Type

In STOM, we extend ObjectStore's metatype hierarchy to support spatio-temporal object types. <Fig. 2> depicts the extended type hierarchy for aspatial, spatial and temporal objects. The leaf nodes are directly instantiable.

types to represent sets of intervals over the real numbers, over the integers, etc. The range object provides these types.



<Fig. 2> Spatio-temporal object type hierarchy

Every object representing a type is an instance of a subtype of the metatype os_type in ObjectStore. To extend metatype hierarchy, we add five new types : graphic_type, geo_type, temporal_type, time_type and range_type. Graphic_type represents things for graphic display. Geo_type represents various vectorized objects such as points, lines, and various type of areas. A point is the values of n coordinates in n-dimensional vector space. A line is a finite sequence of straight line segments. An area is a polygon and it may have holes. Temporal_type represents temporal type which corresponds to geo_type and graphic_type. The temporal types obtained through the moving_type and moving_type represents moving point(mpoint), moving line(mline), and moving area(marea). The intime object converts a given type x into a type that associates instant values with values of x. Time_type represents time which is considered to be linear and continuous. And instant represents time points. Range_type represents

3.2 Spatio-temporal Operator

We give spatio-temporal operators on sorts.

- *Location operators* are the most basic operators for the spatial object processing. They search the location value of the spatial object and retrieve the coordinate of the object, or the boundary of the object.

- *Arithmetic operators* perform arithmetical measurements for spatial objects and have os_integral_type as their return type. An example would be the calculation of the distance between two geo_types.

- *Topological relationships* are the most fundamental operators on geo_type. Topological relationships of two spatial objects are DISJOINT, MEETS, EQUALS, INTERSECTS, COVERS, COVEREDBY, INSIDE, and OUTSIDE.

· *Semantic spatial relationships* are semantic relationships among spatial objects. Semantic spatial relationships of two spatial objects are NEARBY and FAR_AWAY.

· *Mouse operators* provide the function of selecting an object from the previous query result using the mouse. Examples of the mouse operators are pick(), window(), point(), line(), and circle().

· *Object construction operators* are defined to construct new spatial objects which satisfy certain topological relationships. If the relationships do not hold, the operator returns an empty spatial object. Examples of the object construction operators are split, merge, difference, and intersection

· *Time projection operators* are offered to return the parts of the projections corresponding to moving and intime object types. Examples of the time projection operators are deftime, rangevalues, inst, and val.

· *Time intersection operators* concern intersections with the time object and range objects. Examples of the time intersection operators are atinstant, atperiods, initial, final, present, at, atmin, atmax, and passes.

· *Time lifting operators* uniformly lift operations on non-temporal types to the corresponding temporal types. The new operations are given the same name as the operation they originate form.

· *Time changing operators* provide an important property of any time-dependent value which is its rate of change. Examples of the time changing operators are derivative, speed, turn, and velocity.

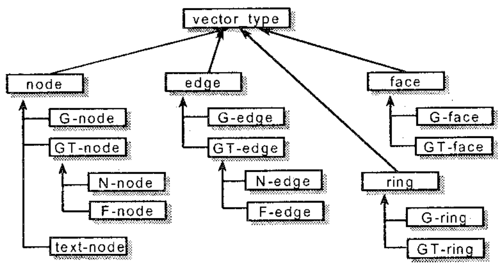
4. Physical layer

In this subsection, we give the physical layer. The physical layer is represented by the spatio_temporal internal description model (STIM). STIM consists of two parts : spatial internal description model and temporal internal description model. Spatial internal description model supports internal vector map structure for spatial type with various levels of topology. Temporal internal description model supports sliced representation for temporal type.

4.1 Spatial Internal Description Model

Maps are the basic internal objects in geographical database system. The spatial internal description model is on a basis of the multiple map layers whose type is a multiple-valued vector map[5]. Using multiple map layers technique for spatial internal structure is nature to organize data and is efficient in terms of object manipulation and storage. Multiple map layers with a multiple-valued vector map structure represent several thematic objects which are captured by STOM. Each layer consists of the set of modules and maintains the topological integrity. An efficient and appropriate data model to handle geographical locations and spatial relationship between objects are required to manage spatial object. That is very important since a poor internal structure would strongly affect response time to queries. For this reason, we have considered three topological levels (geometry, network topology, and full topology level) that may match the user's need.

Each module consists of the set of spatial internal primitive objects which is the lowest internal object. <Fig. 3> depicts the type hierarchy for the spatial internal primitive objects. The end nodes of the hierarchy are instantiated with objects.



<Fig. 3> Spatial internal primitive object type hierarchy

All of the spatial internal primitive objects shown in <Fig. 3> are discussed in details. Node type represents a zero dimensional object that specifies geometric location, or geometric location and the topological connectivities of one or more edges. G-node type represents an internal or extreme point of a line, or not a topological junction of two or more lines. GT-node type represents a point used for identifying the location of point features (or areal features collapsed to a point), such as towers, buildings, places, buoys, etc., and a zero-dimensional object that is a topological junction of two or more links or chains, or an end point of a link or chain. F-node type represents a node which is contained in a single face only. N-node type represents a zero-dimensional object that is a topological junction of two or more lines. Text node type represents a reference point used for displaying map and chart text (e.g., feature names) to assist in feature identification. G-node type represents an internal or extreme point of a line, or not a topological junction of two or more lines.

Edge type represents a connected non-branching sequence of line segments specified as the ordered sequence of points between those line segments. G-edge type represents a sequence of G-nodes. GT-edge type represents a sequence of one or more N-nodes and zero or more G-nodes. N-edge type is an edge which has N-node as start node and end node. F-edge type is a N-edge which has information about left and right faces.

Ring type is a sequence of nonintersecting edges with closure. A ring represents a closed boundary, but not the interior area inside the closed boundary (for full topology level only). G-ring is a sequence of closed G-nodes. GT-ring is a sequence of closed F-edges.

Face type represents the area which is constituted closed boundaries (for full topology level only). Face has one outer boundary and zero or more inner boundaries. G-face type represents a face which has G-ring boundaries. GT-face type represents a face whose boundaries are made up GT-rings.

Using the spatial internal description model, we are able to manage concurrently three different levels of topology. So user can match topological levels depend on his necessity. The definition of primitive objects depends on topological levels.

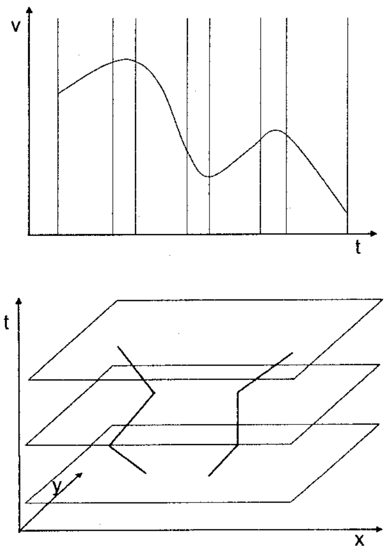
· *Geometry level* describes only geographical location information of each spatial object, but no topological information is explicitly present. Area information may be captured in geometry level by the use of closed lines (G-ring) which circumscribe an area and G-face. Primitive objects are text node, G-node, G-edge, G-ring, and G-face.

· *Network topology level* describes locational information as well as edge to node topological relationships, where every edge must begin and end on a node; however, edges may cross. This level of topology is sufficient to describe connectivity. Planar graph introduces the additional mathematical constraint that edges may not cross except at a node. This permits adjacency to be calculated although it is not directly stored in the structure. Primitive objects of network topology level are G-node, text node, N-node, and N-edge.

· *Full topology level* introduces the concept of a face and describes face to edge as well as node to face topological relationships and locational information. Primitive objects of full topology level are N-node, F-node, F-edge, GT-ring, and GT-face.

4.2 Temporal Internal Description Model

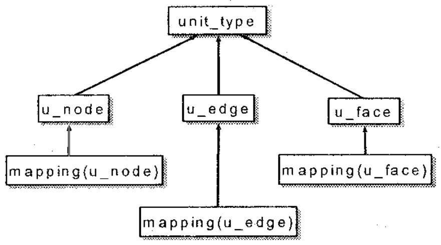
The most important part of temporal internal description model is how temporal types are represented. Temporal internal description model is on a basis of sliced representation[11].



<Fig. 4> Sliced representation of temporal type

The basic idea of the sliced representation is to decompose the temporal evolution of a value into fragments called "slices" such that within the slice this evolution can be described by some kind of "simple" function. <Fig. 4> depicts the sliced representation of temporal types.

The sliced representation is built by a type describing a single slice which we call a unit type(unit_type). A value of a unit type is a pair (i, v) where i is a time interval and v is some representation of a simple function defined within that time interval. We define unit types u_real, u_node, u_nodes, u_edge, and u_face. For values that can only change discretely, there is a trivial "simple" function, namely the constant function. It is provided by a const object type which produces units whose second component is just a constant of the argument type. This is in particular needed to represent moving int, string, and bool values. The mapping data structure basically just assembles a set of units and makes sure that their time intervals are disjoint. <Fig. 5> depicts the type hierarchy for the temporal internal primitive objects.



<Fig. 5> Temporal internal primitive object type hierarchy

5. Spatio-temporal Query Language

A query consists of two parts : the target part specifying the required information and the qualification part specifying the conditions the information to be queried must satisfy. In a spatio-temporal query, it is important to display of query results in a diverse graphical form.

5.1 Spatio-temporal object-oriented query language

Spatio-temporal object-oriented query language (STOQL) is an OQL-based query language. OQL(object query language)[1] is a SQL-like object oriented query language, and its syntax looks as follows:

```
query ::= SELECT[DISTINCT] query
        FROM identifier IN
                query{, identifier IN query}
        [WHERE query]
```

The most obvious difference between conventional and spatio-temporal systems is the ability to spatial data graphically[3]. Interactive-graphic presentation is powerful for mapping systems, because the content of maps can be quickly modified. Objects can be added to, removed from, or modified on an existing map without the need to start with a new drawing from scratch again. This implies that the user must have tools to manipulate a map. Unlike conventional systems which treat each result as an entirely new representation and do not refer to earlier results, several related graphic drawings are often overlayed over each other or on 'conceptual layer' is removed from another. These processes are the particular power of spatio-temporal database systems. Furthermore, individual objects on a map may be highlighted to facilitate their identification in complex situations. Hence, query language should support various presentation types for maps.

For providing an integrated mechanism for the graphical display of spatio-temporal objects and the retrieval of objects are executed, we add DISPLAY-ON-STORE clause in OQL syntax. The STOQL has the following structure:

```
[DISPLAY <display-form-comma-spec-list>]
[ON <layer-name-comma-spec-list> :
        <display condition>]
[STORE <temporary-storage-name>]
OQL statement
```

The DISPLAY clause allows users to tailor the graphical display of query results to their specific needs. To fulfill this, STOQL supports various display forms (COLOR, REMOVE, BLINK, FILL, and SYMBOL) in <display-form-comma-spec-list>. To specify display forms on spatio-temporal objects, users select the current settings using the full-down menus for the setup. The set of display forms is extensible to apply different applications.

The ON clause is used in order to display query results on a layer of the base map. The <layer-name-comma-spec-list> represents the base layer where spatio-temporal objects are displayed, and <display condition> takes one of the following two display modes: (1) NEW clears the viewpoint before drawing the result maps; (2) OVERLAY overlaps the current query result on an existing maps. When the ON clause is omitted, the query results are displayed on the layer where the spatio-temporal objects are stored.

The STORE clause is used to store the query results in the specific storage name <temporary-storage-name> in order to use previous query results in the following query. The <temporary-storage-name> represents the name of the temporary storage named 'temp_sys'.

5.2 Query Examples

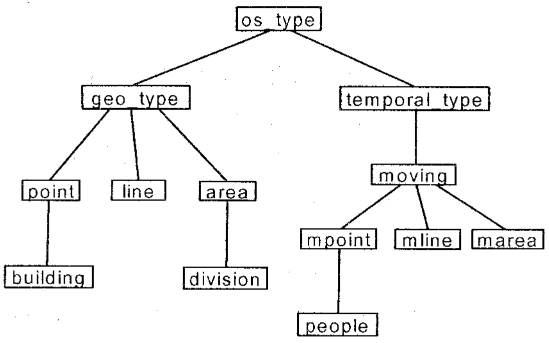
In this subsection, we first give a number of spatio-temporal query examples and show how theses spatio-temporal query are expressed with STOQL by different operators.

Assume a spatio-temporal database contains the following classes :

```
· class geo_type {
      name : string;
  public:
      virtual real *location( );
  }
· class point : public geo_type {
      node *n;
  }
· class line : public geo_type {
      edge *e;
  }
· class area : public geo_type {
      face *f;
  }
· class building : public point {
      owner : string;
      address : string;
      purpose : string;
  }
· class division : public area {
      classification : string;
      population : integer;
  }

· class moving {
      name : string;
  public:
      virtual real *location( );
      virtual real *trajectory( );
  }
· class mpoint : public moving {
      u_node *un;
  }
· class mline : public moving {
      u_edge *ue;
  }
· class marea : public moving {
      u_face *uf;
  }
· class people : public mpoint {
      activity : string;
  }
```

<Fig. 6> shows the class hierarchy of the above database.



<Fig. 6> Class hierarchy of the example

The following queries are examples of STOQL.

Query1 : Where is Minsu at 10 o'clock?

```
SELECT p.location
FROM p IN people
WHERE p.name = "Minsu"
AND p.trajectory = 10
```

Query2 : When does Minsu stay at the "Kangnum-ku"?

```
SELECT p.trajectory
FROM p IN people
WHERE p.name = "Minsu"
AND p.location IN
(SELECT d.location
FROM d IN division
```

```
  WHERE d.classification = "ku"
  AND d.name = "Kangnum-ku" )
```

Query3 : Where is Minsu while Sueyon is at the wedding hall in "Dobong-ku"? Display Minsu in blue color on division maps. And store the result in temp_1.

```
  DISPLAY COLOR(P, "BLUE")
  ON division : NEW
  STORE temp_1
  SELECT result(P : p2.location)
  FROM b IN building, p1 IN people, p2 IN
people
  WHERE p1.name = "Sueyon"
  AND p2.name = "Minsu"
  AND b.purpose = "wedding hall"
  AND p1.location = b.location
  AND p1.trajectory = p2.trajectory
  AND b.location  IN
  (SELECT d.location
  FROM d IN division
  WHERE d.classification = "ku"
  AND d.name = "Dobong-ku" )
```

6. Conclusion

In this paper, we have presented spatio-temporal data modeling and language for an object-oriented spatio-temporal database system. For handling the structure of spatio-temporal objects and the approximate spatio-temporal operators, we have proposed the two layers of object-oriented data models: a spatio-temporal object model (STOM) for the conceptual layer and an spatio-temporal internal description model (STIM) for the physical layer.

STOM supports spatial and temporal object types and their polymorphic operations. To make the spatial and temporal sorts of the STOM, five new object types : geo_type, graphic_type, temporal_type, time_type and range_type, have been added in the ObjectStore's metatype hierarchy. Also several polymorphic operations have been defined on spatial and temporal sorts such as location operations, arithmetic operations, topological relationships, semantic spatial relationships, mouse operations, object construction operations, time projection operators, time intersection operators, time lifting operators, and time changing operators. Using this model frame, it is possible to define additional types and operators depending on its applications, which makes the system extensible. STIM supports efficient and appropriate framework to store spatial coordinates and topology between two spatial objects. And also supports sliced representation to decompose the temporal evolution.

A spatio-temporal object-oriented query language should provide graphical display of spatial objects in query results interactively as well as handing temporal objects. And spatial object type values to be used as "constant" in queries using pointing device such as mouse. For providing an integrated mechanism for the graphical display of spatial objects and the retrieval of spatio-temporal objects, STOQL was designed. This gives the unified view for spatial, temporal and aspatial objects.

We will implement our model and language later on.

References

[1] R. Cattell, D. Barry(1997), The Object Database Standard : ODMG-93, Morgan Kaufmann.

[2] DGIWG(1994), DIGEST(digital geographic information exchange standard), edition 1.2, Defence Mapping Agency, USA, Digital Geographic Information Working Group.

[3] M. Egenhofer(1994), Spatial SQL: A Query

and Presentation Language, *IEEE Transaction On Knowledge and Data Engineering*, Vol.6, No.1, pp.86-95.

[4] M. Erwig(2004), Toward Spatio-Temporal Patterns, *Spatio-Temporal Databases Flexible Querying and Reasoning*, Springer-Verlag, pp.29-53.

[5] B. Falcidieno, C. Pienovi, and M. Spagnuolo(1992), Descriptive modeling and prescriptive modeling in spatial data handling, *Proceedings of the International Conference GIS-From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning*, Pisa, Italy, pp.21-23, LNCS Vol.367, New York: Springer-Verlag, pp.122-135.

[6] Spatial Data Transfer Standard (SDTS) (1992), Federal Information Processing Standards Publication.

[7] L. Fegaras, C. Srinivasan, A. Rajendran, D. Maier(2000), Lambda-DB: An ODMG-Based Object-Oriented DBMS, *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Dallas.

[8] S. Grumbach, et al.(2003), Spatio-temporal Models and Languages : An Approach Based on Constraints, *Spatio-Temporal Databases The CHOROCHRONOS Approach, LNCS Vol. 2520*, Springer-Verlag, pp.177-201.

[9] R. Guting(1993), Second-order signature: A tool for specifying data models, query processing and optimization, *Proceedings of the ACM SIGMOD Conference*, pp.277-286.

[10] R. Guting(1994), An Introduction to Spatial Database Systems, *VLDB Journal*, 3, pp.357-399.

[11] R. Guting, et al.(2003), Spatio-temporal Models and Languages : An Approach Based on Data Types, *Spatio-Temporal Databases The CHOROCHRONOS Approach, LNCS Vol. 2520*, Springer-Verlag, pp.117-176.

[12] T. Hadzilacos and N. Tryfona(1997), An Extended Entity-Relationship Model for Geographic Applications, *ACM SIGMOD Record, 26(3)*, pp.24-29.

[13] Y. Kim(2003), Knowledge-Based Approach for an object-Oriented Spatial Database System, *Journal of Intelligent Information Systems*, Vol.9, No.3, pp.99-115.

[14] C. Lamb, et al.(1991), The ObjectStore database system, *Communications of the ACM*, Vol.34, No.10, pp.50-63.

[15] G. Langran(1993), Time in Geographic Information Systems, Taylor & Francis.

[16] C. S. Parent, et al.(1999), Spatio-Temporal Conceptual Models : Data Structures + Space + Time, *Proceedings of the 7th ACM GIS*, pp.26-33.

[17] N. Tryfona and C. S. Jensen(2000), Using Abstractions for Spatio-Temporal Conceptual Modeling, *Proceedings of the 2000 ACM Symposium on Applied Computing*, Italy.

[18] M. Vazirgiannis and O. Wolfson(2001), Spatiotemporal Model and Language for Moving Objects on Road Networks, *Advances in Spatial and Temporal Databases 7th International Symposium, SSTD 2001*, Redondo Beach, CA, USA, July 12-15, *LNCS Vol. 2121*, Springer-Verlag, pp.20-35.

김 양 희

1982 이화여자대학교 수학과 (이학사)
1984 서울대학교 수학과 (이학석사)
1988 미국 위스콘신대학교 (메디슨 소재) 전자계산과(이학석사)
1989 미국 위스콘신대학교(메디슨 소재) 전자계산학과 (박사과정 이수)
1997 고려대학교 전자공학과 (공학박사)
1990~1997 수원과학대학 전자계산과 교수
1997~2001 한세대학교 컴퓨터정보통신공학부 교수
2001~현재 한국체육대학교 교양과정부 교수

관심분야 : 시공간 데이터베이스, 객체지향 데이터베이스, 데이터베이스 보안 등
E-Mail : yangh-kim@hanmail.net
yangh-kim@knus.ac.kr