An Overview of Object-Oriented Frameworks with Application in Spatiotemporal Data Mining

K.Venkateswara Rao[1], A.Govardhan[2] and K.V.Chalapati Rao[3]
[1] CVR College of Engineering, Department of CSE, Ibrahimpatan, R.R.District, A.P., India
Email: kvenkat.cse@gmail.com
[2]JNTUH College of Engineering, Department of CSE, Jagityala, Karimnagar Dist, A.P., India
Email: govardhan_cse@yahoo.co.in
[3] CVR College of Engineering, Department of CS*E,* Ibrahimpatan, R.R.District, A.P., India
Email: chalapatiraokv@gmail.com

***Abstract*—An object-oriented framework is a reusable software system providing large scale reuse, including reuse of analysis and design. Frameworks offer reuse of its elements through mechanisms like inheritance and composition and avoid development of applications from scratch. Spatiotemporal data mining has wide variety of applications in transportation systems, surveillance applications, geographical systems and environmental systems. So there is a need to develop a framework that takes care of common requirements at analysis and design level so that spatiotemporal data mining applications can be built through software reuse. The main focus of this paper is to provide an overview of object oriented frameworks, capturing requirements, analysis and design of object-oriented framework for spatiotemporal data mining. The process of reusing the framework for application development is also described and results of such application are provided.**

***Index Terms*—Object-Oriented Frameworks, Data mining, Spatiotemporal data mining**

I. INTRODUCTION

An object-oriented framework is a set of collaborating classes, both abstract and concrete, that embodies a reusable design to provide solutions to a family of related problems. It supports reuse at functionality and architecture level [1]. A framework, then, is an incomplete software system in the sense it provides a partial design and implementation for an application in a given domain. Thus a framework is much more than a class library as shown in Fig 1. A class library is a set of classes designed to provide reusable, general purpose functionality. On the other hand, the goal of a framework is to capture a set of concepts related to a domain and the way they interact. In addition a framework calls specific application code by dynamic method binding [2].

A framework provides a basic system model for a particular application domain within which specialized applications can be developed. It consists of already coded, reusable functions called frozen spots which represent the static parts of the framework. The framework also consists of functions that need to be customized to change the behavior of the framework. These flexible elements are called hot spots and they allow the user to adjust the framework to the needs of the concrete application.

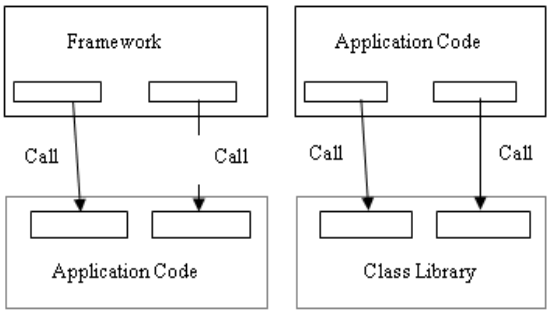

Fig 1: The difference in control between frameworks and class libraries [3].

Spatiotemporal data mining refers to the extraction of knowledge, spatiotemporal relationships, or other interesting patterns like spatiotemporal associations, clusters and evolution rules which are not explicitly stored in spatiotemporal databases. It is an emerging area dedicated to the development and application of novel computational techniques for the analysis of very large spatiotemporal databases. It presents a number of challenges due to complexity of geographic domains and mapping of all data values into spatial and temporal frameworks [4].

The research community in spatiotemporal data mining in different domains is investigating techniques for addressing the issues and requirements of those mining tasks [4]. The research community in spatiotemporal databases could develop conceptual models for spatiotemporal databases, and techniques to implement these models in object-relational and relational database systems [4]. This development enables further investigations in spatiotemporal data mining to discover requirements of various data mining techniques that are developed in various domains and come out an object-oriented framework. The framework help generate various spatiotemporal data mining applications by reusing the architecture and extending functionalities in the framework. The spatiotemporal data mining involves selection, pre-processing and transformation of data, application of the data mining techniques and evaluation and visualization of resulting patterns and trends.

Keeping the above requirements in view, our work presents an overview of object-oriented frameworks and their application in discovering knowledge from spatiotemporal databases. Section 2 presents

 DOI: 10.32377/cvrjst0102

categorization of frameworks based on their scope and context of usage. Different Phases in framework-centered application development are described in section 3. Various activities in the framework development and execution of those activities for spatiotemporal data mining are discussed in section 4. Finally conclusions are given in section 5.

II. FRAMEWORKS CLASSIFICATION

Frameworks are composed of numerous reusable assets which include knowledge, design, captured requirements and software components. As a coherent collection of reusable assets, a framework can be an important investment for an organization. Based on the context of its usage, the frameworks are classified into the following categories.

- White Box Frameworks
- Black Box Frameworks
- Grey Box Frameworks

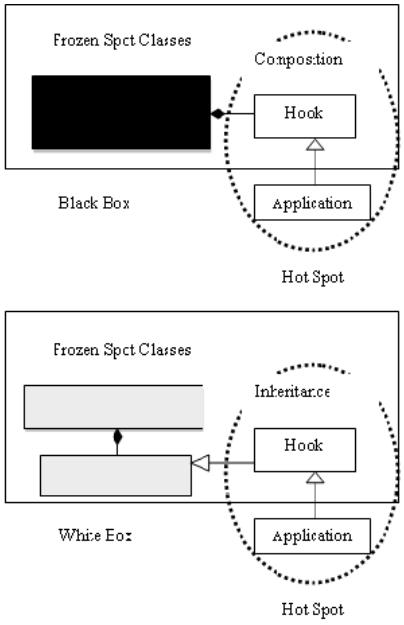

Fig 2: Hot spots in object oriented frameworks: the black box approach and the white box approach [5]

Figure 2. Hot spots in object frameworks: the black box approach and the white box approach [5].

In a white box framework, as shown in fig 2, the architecture of the framework is known and application developers build their applications upon it. The complete design has to be documented because this knowledge is necessary to adapt the framework to a concrete application. The mechanism used to provide flexibility is usually limited to inheritance. The user, therefore, must have knowledge of the framework architecture in order to customize the framework so as to address the needs of the particular application.

Black box framework, as shown in Fig 2, hides its internal structure. The user just knows the hot spots of the framework and a general description of the framework's usage rather than having a comprehensive knowledge of its architecture. Very often the mechanism used to provide flexibility in black box framework is composition. The capabilities of a black box framework are limited to what has been implemented in the set of components provided in it.

Grey Box Framework offers both inheritance and composition mechanisms. It will have a white box layer consisting of interfaces and abstract classes providing the architecture that can be used for white box reuse. It has a black box layer too consisting of concrete classes and components that inherit from the white box layer and can be plugged into the architecture. By using the concrete classes, the developer has easy access to the framework's features. If more is needed than the default implementation, the developer will have to make a custom class either by inheriting from one of the abstract base classes or by inheriting from one of the concrete classes.

Based on the scope of the framework which describes how broad an area of the framework is, they are classified as

- Application frameworks
- Domain frameworks
- Support frameworks

Application Framework is a horizontal framework which covers functionality that can be applied to different domains. Examples of application frameworks are frameworks for graphical user interfaces.

Domain Framework is a vertical framework which captures knowledge and expertise in a particular problem domain. Frameworks for spatiotemporal data mining and manufacturing control are examples of domain frameworks.

Support Framework offers low-level system services such as memory management, device drivers and file access. Other frameworks or applications would be built on this framework.

III. PHASES IN FRAMEWORK DEVELOPMENT

The presence of reusable frameworks influences the development process for the application. There are three phases in framework-centered software development [6].

- framework development phase
- framework usage phase
- framework evolution & maintenance phase

The framework development is the most effort consuming phase and is aimed at producing a reusable design in a domain. Major results of this phase are the domain analysis model and a core framework design. This phase is depicted in Fig 3.

The framework usage phase is also referred as the framework instantiation phase or application development phase. The main result of this phase is an application developed reusing one or more frameworks. Here, the framework user has to include the core framework design or part of it depending on the application requirements. After the application design is finished, the software engineer has to decide which internal increments to include. For those parts of the

application design not covered by reusable classes, new classes need to be developed to fulfill the actual applications requirements. These new classes are referred as the application-specific increment.

Frameworks are typically developed and evolved in an iterative way [7]. Once the framework is released, it is used to create applications. After some time it may be necessary to change the framework to meet new requirements. This process is called framework evolution. Framework evolution has consequences for applications that have been created with the framework. If APIs in the framework change, the applications that use it have to evolve to remain compatible with the evolving framework.

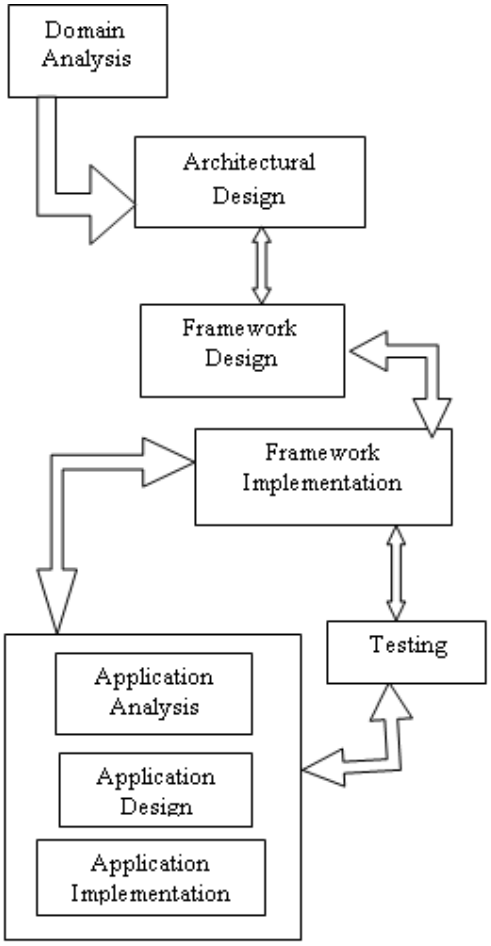

Fig 3: Framework development phase

IV. Spatiotemporal Data Mining Framework Development

The development of a framework is somewhat different from the development of a standard application. The important distinction is that the framework has to cover all relevant concepts in a domain, whereas an application is concerned only with the concepts mentioned in the application requirements. The activities of a simple framework development for spatiotemporal data mining are described in following sections.

A. Domain Analysis

Domain analysis aims at describing the domain that is to be covered by the framework. It captures the requirements and identification of concepts by referring to previously developed applications in the domain, domain experts and existing standards for the domain. The result of the activity is a domain analysis model, containing the requirements of the domain, the domain concepts and the relations between those concepts.

The Literature on spatiotemporal data mining is analyzed and spatiotemporal applications requirements at the modeling level are extended to cover knowledge discovery process [8] as follows.

- Need for representation of objects with position in space and existence in time: An example is "waterpipes" occupying certain parts of space at certain time point in utility management information system.
- The need to capture the change of position in space over time: If the change of object's position is continuous, then it results into motion. An example is the continuous change of the position of a vehicle in a navigational system. If the change is discrete, then it results into snapshots, versions of the object. An example is a "land parcel" having a position in space at some point in time. When it changes shape (e.g., a new part of land is attached to it) its position changes.
- Need for the definition of spatial attributes in time and organizing them into temporal layers or fields, i.e., snapshots of thematic maps. For example, "soil erosion" is a property of space organized in a layer, representing sets of regions (with different values). Spatial attributes can be visualized as continuous (e.g., "temperature") or as discrete ("soil type").
- Need to capture the change of spatial attributes over time: The changes can be discrete (e.g., changes on a map of "land parcels" or "vegetation") or continuous (change of "temperature").
- Need to connect spatial attributes to objects: An example, a land-parcel that has "soil type" as an attribute. The "soil type" is an attribute of space and land-parcels inherit part of it.
- Need for the representation of different spatiotemporal topological relationships among spatial objects in time for analysis.
- Need for the representation of relationships among spatial attributes in time: For example, the "soil type" is a result of the combination of the "acidity" and the "corrosively" of soil.
- Need to specify spatiotemporal integrity constraints: The constraints are imposed either by the user or by the designer for the integrity of the database.
- Need to represent the orientation and direction features of spatio-temporal objects during their change.

- Need to represent events and list of changes associated with each event: An event may represent abrupt change or it may have duration.
- Need for the representation of multiple granularities for spatiotemporal objects: As an example, when tracing modifications to spatial areas, the history of the areas under observation has to be maintained and retrieved at multiple temporal granularities (e.g., years, months, decades).
- Need for representation of spatiotemporal data in multi-dimensional model for analytical processing.
- Need for the representation of concept hierarchies for the dimensions.

A. Architectural Design

In this stage, a suitable architectural style underlying the framework is to be decided upon domain analysis. The architectural design for spatiotemporal data mining framework can be specified using layered pattern and model-view-controller pattern. The four layer architecture specified for information systems in [9] can be used as an underlying architecture for spatiotemporal data mining framework. The layers are described in the following Fig 4.

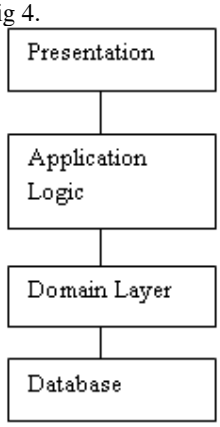

Fig 4: Layered architectural pattern for spatiotemporal data mining

The database layer holds domain specific spatiotemporal data. The domain layer models the conceptual structure of the problem domain in terms of classes and class diagrams. Application logic contains the data mining algorithms and data pre-processing and analysis techniques. Presentation layer is used to show the data mining results to the user and accepting input in interactive data mining.

A model-view-controller pattern [9, 10] can also be used to describe the architecture of spatiotemporal data mining framework. The model holds data in the form of multi-dimensional data cube and contains algorithms to perform appropriate analysis and mining on the data in the multi-dimensional model. The user input comes to the controller. The controller translates the user events into service requests for the model or the view. The view part of the pattern obtains results from the model and displays them to the user. So this pattern spans the presentation, application logic and domain layers of the layered pattern. The MVC is shown in Fig 5.

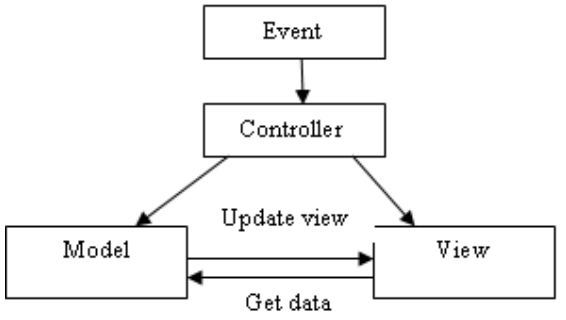

Fig 5: MVC architectural pattern for spatiotemporal data mining

B. Framework Design

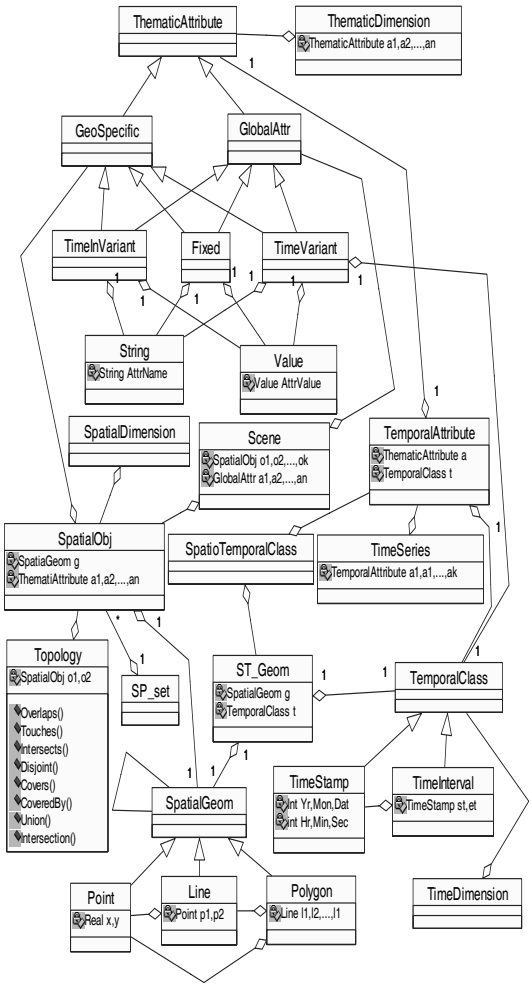

Fig 6a: Class diagrams of the system

The classes in the architectural design of the framework are refined and additional classes are designed during this phase. Results from this activity are the functionality scope given by the framework design, the framework's reuse interface, design rules based on architectural decisions that must be obeyed and a design history document.

The requirements of spatiotemporal applications are analyzed to identify different classes, attributes and methods of each class and relationships among the classes. The results are represented in a class diagram [8] as shown in Fig 6a and Fig 6b. The classes can be extended using inheritance mechanism, while building the applications to meet their requirements.

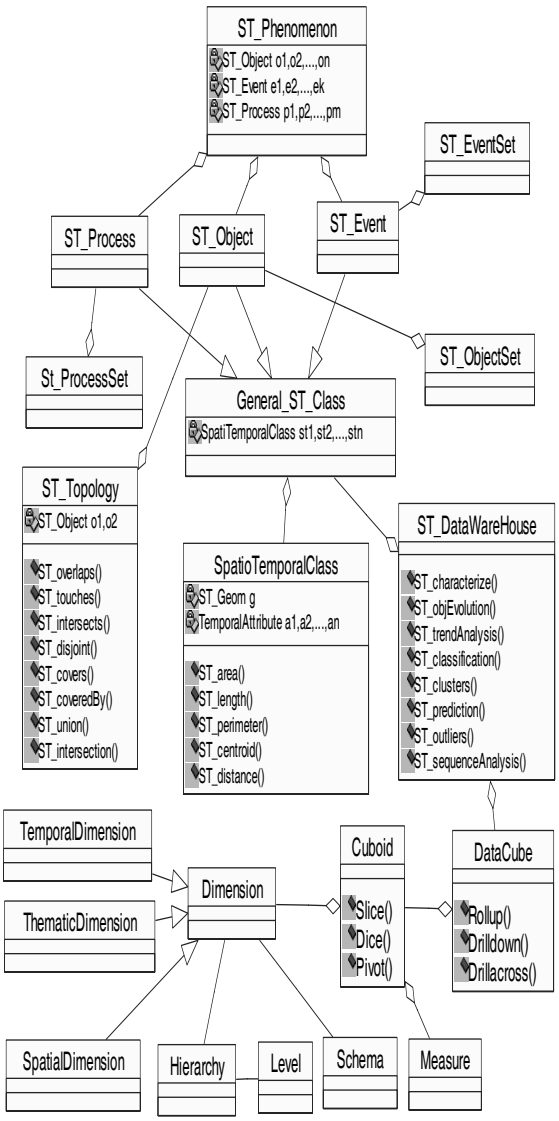

Fig 6b: Class diagrams of the system

An object relational spatiotemporal database which consists of a set of tables and relationships among them is designed to facilitate the spatiotemporal data analysis and mining [11]. Different sorts of spatiotemporal data to be handled are states, events and episodes. A state represents a version of an entity in a given moment. States can consist of different versions of an individual entity. An event is the moment in time when an occurrence takes place. Event causes one state to change to another. An episode is the length of time during which change occurs, a state exists or an event lasts. Two main strategies to represent multiple versions of an object are tracking the versions either at the level of objects or attributes. First one involves a different identifier (oid) to each new version and chaining the older versions to new oid. Second one involves a single object identity (oid) with versions actually associated with attributes. The attributes of spatiotemporal objects can be categorized as version significant, non-version significant and invariant. The version significant attribute values are to be updated in non-destructive manner, the non-version significant attribute values are to be updated in a destructive manner and invariant attributes values are not allowed to be changed. Following entities are designed to address these requirements.

1. Temporal_tab: This table stores timestamps which correspond to the time at which change to any spatial object has taken place.
2. Spatial_obj_tab: This table contains spatiotemporal objects with unique identifier, geometry and existence time which has from time (st) and to time(et) as attributes. It also has other attributes to indicate category and type of spatial object and type of change. The events and episodes or processes are also considered as spatiotemporal objects and stored in this table.
3. split_tab: This composite entity is used to record splitting of any spatial object into multiple objects. It has object identifier that got split and new object identifiers for objects derived due to split and timestamp attribute that records time of split.
4. Merge_tab: This composite entity keeps the data related to merging of two or more objects into a single object. It maintains object identifiers which are merged, new object identifier for object derived due to merging and timestamp attribute which records the time of merge. The new objects created due to split or merge are stored in spatial_obj_tab with their new object identifiers.
5. Geom_version : This table records geometry changes by creating new object identifier for each change to the geometry of the object. It has oid of the object changed, new object identifier and timestamp attribute that records time of the change.The new object is stored in spatial_obj_tab table.
6. VsAttr_tab: This table manages version significant attributes of all objects in spatial_obj_tab. It has oid, attribute name, timestamp and attribute value as its fields.
7. VinAttr_tab: This table manages version insignificant and invariant attributes of all objects in spatial_obj_tab. It has oid, attribute name and attribute value as its fields.
8. Result_tab : This entity is used by analysis algorithms to store results back into database. This table can be accessed using OpenJump (An Open source GIS software) to visualize the results.

C. Framework Implementation

Framework implementation is concerned with the coding of its abstract and concrete classes. The database

designed for spatiotemporal data mining [11] is created and implemented using postGIS, postgresql. The Topology class specified in the class diagram in fig 6 is implemented as GeometryRelation using Postgis application programming interface.

```
Public class GeometryRelation
{
        PGgeometry obj1, obj2;
        Methods:
        PGgeometry Union();
        PGgeometry Intersection();
        Float Distance();
        Boolean isintersects()
        Boolean istouches()
        Boolean isequals()
        Boolean isdisjoint()
        Boolean iscrosses()
        Boolean isoverlaps()
        Boolean iscovers()
        Boolean iscoveredby()
}
```

The ST_Toplogy class is also implemented by using GeometryRelation object in algorithm1 and algorith2 [11] given below.

Algorithm 1: Tracking Spatial object or History Topology
Input: Spatiotemporal dataset (D),Spatial Object Identifier (oid), from time(f_t) and to time(t_t)
Output: The Object (oid) details changing with time.
Method: Process(oid)

```
Begin
  Obj = getobjdetails(oid) /* connect to dataset D get spatial object details and load them into variable  obj */
   Display(Obj)
   If ( Obj.et  <  t_t) Track(Obj)
End

Display(Obj)
Begin
 If ( Obj.st > f_t and Obj.et < t_t )
   Print or save id, geom, st, et, area, centroid and perimeter of Obj. The attribute ChangeType of Obj indicates type of change the Obj has undergone at time et.
 Elseif ( Obj.st > f_t and Obj.et >= t_t )
   Print or save id, geom, st, t_t, area, centroid and perimeter of Obj.
 Elseif ( Obj.st <= f_t and Obj.et >= t_t )
   Print or save id, geom, f_t, t_t, area, centroid and perimeter of Obj.
 Elseif ( Obj.st <= f_t and Obj.et < t_t )
   Print or save id, geom, f_t, et, area, centroid and perimeter of Obj. The attribute ChangeType of Obj indicates type of change the Obj has undergone at time et.
 Else Print " Obj is not valid between f_t and t_t"
End

Track(Obj)
Begin
Next = Obj.changeType
Switch(Next)
Begin
Case C:
        Look into Geom_version table and find new object identifier (n_oid).
        Process(n_oid)
        Break;
Case S:
        Look into split_tab and find list L of identifiers of objects which are result of split of the Obj.
        For each object identifier e_oid in L, Process(e_oid)
        Break
Case M:
        Look into Merge_tab and find the new object identifier (n_oid) which is the result of merge of Obj with some other spatial object.
        Process(n_obj)
        Break
Case default: break
End
End
```

Algorithm 2: Finding change in spatiotemporal topological relationships, intersection, Union, distance between two objects.
Input: : Spatiotemporal dataset (D), Object Identifier (oid1, oid2), from time(f_t) and to time(t_t)
Output: Topological relationships, Intersection, Union, distance changes between the oid1 and oid2 from f_t to t_t
Method :

```
Begin
```

1. Using Algorithm1, track the objects oid1, oid2 and record all time points at which either oid1 or oid2 or their siblings have changed between f_t and t_t. Also use special data structure that manages valid identifiers of the objects for each of the time points.
2. For each of the time points, create an object of type GeometryRelation and use its methods to compute topological relationships, intersection, union and distance between the relevant pair of objects which are valid for the time point.
3. Display or store the results for the given duration.

```
End
```

D. Framework Testing

Framework testing is performed to determine whether the framework provides the intended functionality and also to evaluate the usability of the framework. The only way to do this is to reuse it. So this boils down to developing applications that use the framework.

To evaluate the usability of spatiotemporal data mining framework, the test application based on the framework is developed for discovering spatiotemporal topological relationships and spatiotemporal frequent itemsets. The results are found to be satisfactory. But the framework implementation and testing need to be done for other spatiotemporal data mining tasks such as clustering, classification, characterization and discrimination.

E. Test Results

The results of test application to discover spatiotemporal topological relationships are given below. Given two objects 45 and 83, and two timestamps as input, spatiotemporal relationships among the given objects over the specified period of time are indicated in the following output. The object changes are tracked by creating a new object in database for each change to its geometry.

Obj1	Obj2	intersects	contains	equals	touches	disjoint	overlaps	From time	To time
45	83	0	0	0	0	1	0	T1	T2
45	85	0	0	0	1	0	0	T3	T4
45	95	0	0	0	0	0	1	T5	T6
54	95	0	0	0	0	1	0	T7	T8

T1=2000-01-08 01:10:15 T2=2000-01-08 09:10:15
T3=2000-01-08 09:15:15 T4=2000-01-09 01:10:15
T5=2000-01-09 01:15:15 T6=2000-01-10 01:10:15
T7=2000-01-10 01:15:15 T8=2000-03-10 01:10:15

CONCLUSIONS

In this paper, object oriented frameworks are described along with their classification. Various phases of the framework development are elaborated. The framework development methodology is applied to spatiotemporal data mining. The requirements of the spatiotemporal data mining are analyzed and suitable architecture for object oriented framework is identified and described using layered and model-view-controller architectural patterns. The framework structure is represented using class diagrams. The generic design for spatiotemporal database is also discussed. The framework is applied to generate applications for discovering spatiotemporal topological relationships and association patterns. The results of the topological relationships application are discussed. However, the framework needs to be extended to support other spatiotemporal data mining applications for association analysis, cluster analysis and classification.

REFERENCES

[1] Savitha Srinivasn, Design Patterns in Object-Oriented Frameworks, IEEE Computer, February 1999,24-32.

[2] Michel Jaczynski and Brigitte Trousse, An Object-Oriented Framework for the Design and the Implementation of Case-Based Reasoners, 6th German Workshop on Case-Based Reasoning, 6th-8th, March,1998, Berlin, Germany.

[3] Niklas Landin and Axel Niklasson, Development of Object-Oriented Frameworks, CODEN: LUTEDX (TETS-5231) /1-146/(1995)

[4] K.Venkateswara Rao, Dr.A.Govardhan, and Dr.K.V.Chalapati Rao, A Generic Framework for Spatio-Temporal Data Mining System, APSMS-JMS, July 2008.

[5] David Parsons, Awais Rashid, Andreas Speck and Alexandru Telea, A"Framework" for Object Oriented Frameworks Design, Proceedings of TOOLS '99', IEEE Computer Society,1999, 141-151.

[6] Jan Bosch, Peter Molin, Michael Mattsson and PerOlof Bengtsson, Object-Oriented Frameworks-Problems & Experiences, Research Report 9-97, ISSN 1103-1581, BIT, Sweden.

[7] J.van Gurp and J. Bosch, Design, Implementation and evolution of object oriented frameworks: concepts and guidelines, Software- Prectice and Experience (SP&E), 2001;31;277-300.

[8] K.Venkateswara Rao, A.Govardhan and K.V.Chalapati Rao, An Object-Oriented Modeling of Spatio Temporal Knowledge Discovery System, ICACC-2011, NIT Hamirpur.

[9] Frank Buschmann etal, Pattern-Oriented Software Architecture: A System of Patterns, Wiley India, 25-51,125-144.

[10] Weiwen Yang, Yanzhen Qu and Richard fairley, improving the data warehouse Architecture Using design Patterns, Proceedings of the Sixth Midwest Association for Information Systems Conference, Omaha, NE May 20-21, 2011.

[11] K.Venkateswara Rao, Dr.A.Govardhan and Dr.K.V.Chalapati Rao. Discovering Spatiotemporal Topological Relationships, DMS-2011, July 15-17, Chennai.