American Journal of Applied Sciences 2 (12): 1584-1594, 2005
ISSN 1546-9239

Construction of an Integrated Object Oriented System for Temporal GIS

[1]Souheil Khaddaj, [1]Abdul Adamu and [2]Munir Morad
[1]Faculty of Computing, Information Systems and Mathematics
[2]School of Earth Sciences and Geography, Kingston University
Kingston upon Thames, Surrey KT1 2EE, UK

Abstract: The development of temporal GIS systems capable of dealing with the evolution of geographical objects is still a challenging task. Although many successful GIS systems have been implemented, comparatively little has been done on how to determine and analyze the patterns of continuous changes of geographical objects. The aim of this paper is to construct a GIS system using an object-oriented model, which is able to track the evolution of geographical objects over time continuously. In addition, the system can allow the events and processes related to each object to be determined and analyzed.

Keywords: Object oriented modelling, object versioning, object database, GIS

INTRODUCTION

The current GIS approaches for representing changes of geographical phenomena do not provide the ability to examine the complexities within these changes, for different themes (e.g. social activities, infrastructure facilities, transportation, population etc). This limitation has become a major problem because a change in one theme may have an adverse effect on others, thereby impeding the work of urban planners, for instance, who need to monitor not only changes but the interdependencies between these. This situation is exacerbated by the fact that no adequate data models are available which could efficiently represent detailed changes, showing the pattern of relationships among the themes, the cause of changes and the result of the changes over time[1].

The aim of this research is to develop an integrated temporal GIS system using an object-oriented model (OOM) combined with an object-oriented programming environment (OOPE) and an object-oriented oriented database system (OODBS). The system is able to track continuously the evolution of geographical objects using contemporaneously the temporal relationships between the versions of the geographical objects. In addition, the system is able to allow the events and processes related to each object to be determined and analysed. The OOM used in the implementation is a conceptual model that represents the semantics to allow the continuity and pattern of changes of the geographical objects to be determined based on the temporal relationships between the versions, events and the processes over a time period[2].

The OODBS used is Objectivity database (Objectivity 2000) implemented using visual C++ environment. The versioning mechanism of the database allows the versions of a geographical object to be related, thereby creating temporal relationships between them. The implemented GIS system provides an optimum query mechanism by enabling continuous forward and backward tracking movements. For example, the previous version and next version of a geographical object can be retrieved from the current version.

In this work, we start by considering the development of a model to determine the continuous links between different versions of geographical objects and the attribute changes such as temporal, spatial and thematic. Then, temporal relationships between the versions, events and the processes are presented. An object oriented tool, the Unified Modelling Language (UML) is then used to describe the design of the conceptual model, followed by an evaluation of the implementation used to achieve the design objectives. Finally, we present some conclusions and suggestions for future work.

OBJECT ORIENTATION AND GIS

Current modelling of spatio-temporal changes covers a wide spectrum of applications, such as socio-economic analysis, environmental impact assessment, epidemiological projections and transportation planning. Different types of data models, based on raster and vector approaches[3], have been used to represent changes of geographical phenomena.

Relational models have been found to be well suited for applications where the relationships among the geographical objects are fairly fixed and well-known. By contrast, object-oriented models can outperform relational models at handling complex relationships among the geographical objects and in

Corresponding Author: Souheil Khaddaj, Faculty of Computing, Information Systems and Mathematics, Kingston University, Kingston upon Thames , Surrey KT1 2EE, UK., Tel: +44 20 8547 2000, Fax: +44 20 8547 7972

dealing with temporal issues[4,5] and optimum query mechanisms can also be produced by using object-oriented approaches[6]. The need to adopt object oriented approaches have been prompted by the fact that there are problems with relational models in dynamic environment where changes are fast and the databases cannot be redesigned to rapidly deliver the necessary information[7,8]. An object oriented database could model the changes and is based on a mix of geographical objects and their relationships. For example, if a road is represented as an object rather than as an entry in a database table, associations with other objects (e.g. street, building etc) linked to the road can automatically "inherit" any changes made to the road, thereby making it easier to track later[9].

Thus, recent research proposals have used temporal GIS and object oriented techniques to explicitly define the relationship between the events (or processes) and the objects with time. These proposals include the triad model and the event-oriented model, where object oriented techniques have been used to identify the pattern of changes (events) within the objects[6,9]. In order to effectively track versions of the original object version management[5,10] and identity-based methods[11] can be used.

The triad model[12] is an integrated model and consists of three independent and interrelated domains (i.e. location, feature and time); while in event-oriented models[13-15] changes are time-stamped as a sequence of events through time and stored in increasing order from the initial event. Frank[13] represented an ordinal model where events (not time-stamped) are linked in sequential order. Claramunt and Theriault's investigation, however, describes object changes in the past, present and future[16].

Although, the triad model represents an integrated approach for representing changes, it does not relate events to the geographical phenomena, while event-oriented models are suitable for temporally stable changes but are not useful for representing sudden changes (e.g. earthquake) and gradual changes (e.g. rainfall)[17]. The Claramunt and Theriault model[16] adequately represents the events related to the change but stores the changes as attributes of the object and the model is not suitable for tracking the evolution of geographical objects involving splitting, merging or transition.

OBJECT ORIENTED MODEL FOR GIS

The object-oriented approach has the abstraction power to represent real objects, provides the extensibility needed to create new geographical models (through "inheritance"), the semantic needed to construct complex objects of similar spatial and temporal states (through "polymorphism")[4]. The attributes and behaviour are "encapsulated" within the objects, therefore providing a network of relationship with other objects, identifies the pattern of changes within the objects or predicting the effects of the changes[6].

Thus, the proposed model is based on object oriented techniques and it will be referred to as Object Oriented model (OOM) for GIS. The model supports both object and attributes versioning. According to the model, changes of geographical phenomena are handle by version management. A version of the object consists of composite classes as in Fig. 1. The aggregated composite classes include thematic class, spatial class and temporal class. The associated composite classes include events class and processes class. The spatial class deals with queries about the location of the object (e.g. where is the best museum in this city?). The thematic class deals with queries about the features of an object (e.g. what is the highest building? or what is the speed limit of this road?). The temporal class deals with queries about the time attributes of the object (e.g. when was the first hospital built in this locality?). Furthermore, an event class deals with the cause of the changes of the geographical object (e.g. why did they reduce the speed limit of this road?). And a process class deals with effect of the changes of the object (e.g. how much of this rain will cause a flood?).

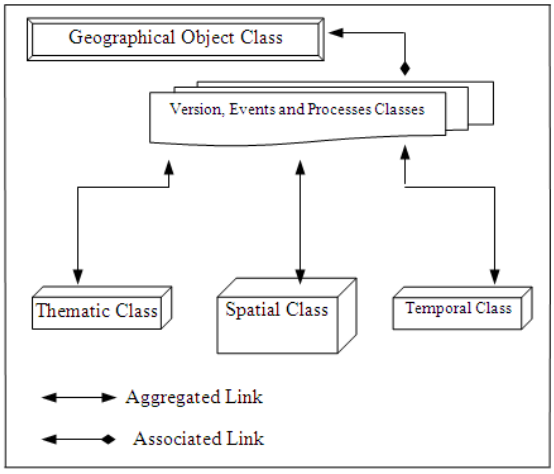

Fig. 1: Composite classes of a geographical object

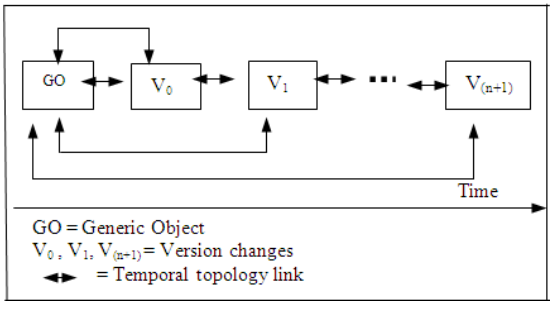

Fig. 2: Relationship between the versions and generic object

The versions class: As Fig. 2 illustrates, a geographical object is represented as a generic object, the first object and any subsequent changes can be represented as versions. Each version of the object consists of changes (involving an attribute or behaviour) of the aggregated classes (spatial, thematic and temporal) and the associated class (events and processes). Subsequent changes of attributes of the versions will generate related dynamic attributes and temporal links to be updated to the respective versions[18]. The relationships between the generic object and the versions of the object are represented by temporal version management approach[5,19]. The version management uses temporal operators (e.g. during, after, before etc) to handle gradual and sudden changes[20]. To avoid the use of large storage space, only the generic object or the current object holds the complete attributes and behaviour of the object whiles the other versions represents the changes of their attributes and behaviour.

The temporal relationships between the current object and versions is given by:

$$\text{Versions}(x) = (\Delta_x (n,n-1), \Delta_x (n-1,n-2),\ldots, \Delta_x (n_0+1, n_0), CV_x(n_0)) \quad (1)$$

Where $CV_x(n_0)$ is the complete version of object x whiles n_0 indicates the generic version, which holds the complete attributes and behaviour. $\Delta_x (k, k')$ represents the difference between the current version (k) and the previous version (k') of object x. As shown in equation (1) access to the current version n requires $n-1$ iterations, which means evaluating delta version $\Delta_x (n_{0+1}, n_0)$ followed by delta version $\Delta_x (n_{0+2}, n_{0+1})$, then the next version up to delta version $\Delta_x (n, n-1)$. This forward oriented versioning strategy equation provides faster access time for the oldest version.

As shown in equation (2), previous versions can be evaluated from current versions and this strategy is known as the backward based versioning. The method in equation (2) provides a quicker access to the current versions.

$$\text{Versions}(x)=(CV_x(n),\Delta_x(n,n-1), \Delta_x(n-1,n-2),\ldots,\Delta_x(n_0+1, n_0)) \quad (2)$$

When a geographical object changes, the generated dynamic attribute locates the versions and creates temporal links between the previous version and the new versions. Similar equations, which can be used for splitting and merging of objects, are given in Dadam *et al.* [19].

Spatial, thematic and temporal classes: A version of an object consists of changes (attribute or behaviour) of the spatial, thematic and temporal classes. Subsequent attributes and behaviour of the classes are automatically updated to the respective class. Each attribute or behaviour change is contained in a version, linked bi-directionally to the respective spatial, thematic and temporal classes. Also, the attribute and behaviour changes of the versions of the object are linked respectively to the previous and next changes. The relationships between the attribute and behaviour changes are linked through the versions by the version link (↔V). Spatial attributes and behaviour changes of the versions have relationships through the spatial links (↔S). Similarly thematic and temporal attributes and behaviour changes of the versions have relationships using the thematic link (↔H) and temporal links (↔T) respectively. The relationships between the attribute and behaviour changes improve the efficiency of the query mechanism of the model and reduce the data access time (Fig. 3). They record only the attribute or behaviour changes to minimise redundancy and data storage. To promote detailed and continuous analysis of the changes, there is relationships (bi-directional) between the changes (attributes and behaviour) of each version.

Events and processes classes: Events are regarded as communication between two or more objects that causes an alteration of attributes or behaviour of any of the objects. The data about a specific event linked to the geographical object come from an external source and are handled by the event class. The event class has an initial associations link with the version class. Subsequent association links between the event class and the version class are produced through the bi-directional links. In view of this, the model should be able to deal with attribute changes (dynamic attributes) and to handle both static and dynamic changes. Recording of events are time-stamped as both absolute time and relative time, in order to be able to deal with known dates of events; but the recording can also be sequential using temporal relationships to determine unknown dates of events. The recording of the attributes of the events (e.g. starting time, ending time, speed, position etc.) enable the model to deal with sudden changes (such as an earthquake for example) as well as gradual changes (such as rainfall).

As shown in Fig. 4, there are temporal relationships between the first events and the subsequent events and between the current event, the previous events and the next event. Also, there is a relationship between different types of events related to the same version of a geographical object (Fig. 4).

The processes have attributes such as start time and end time to be able to handle sudden changes (such as structure damage) as well as gradual changes (such as urban flooding). In certain situations an event will generate more than one process. Also, different types of events related to the same version of geographical object can generate specific processes. In such a case, there are temporal relationships between set of processes. The temporal relation between versions and processes of a geographical object is illustrated in Fig. 4.

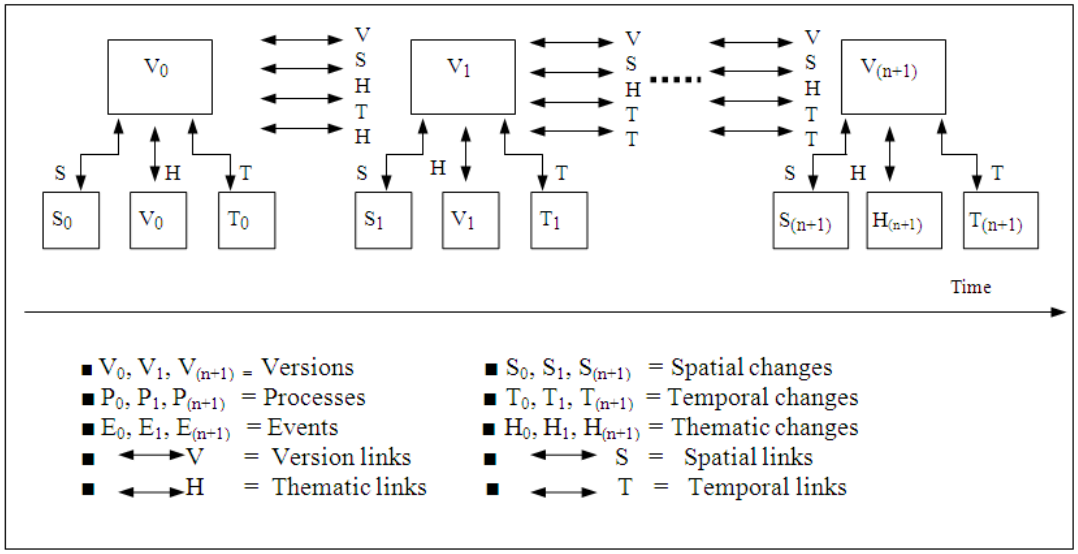

Fig. 3: Relationships between the attributes and behaviours changes of the spatial, thematic and temporal classes of the versions of the object

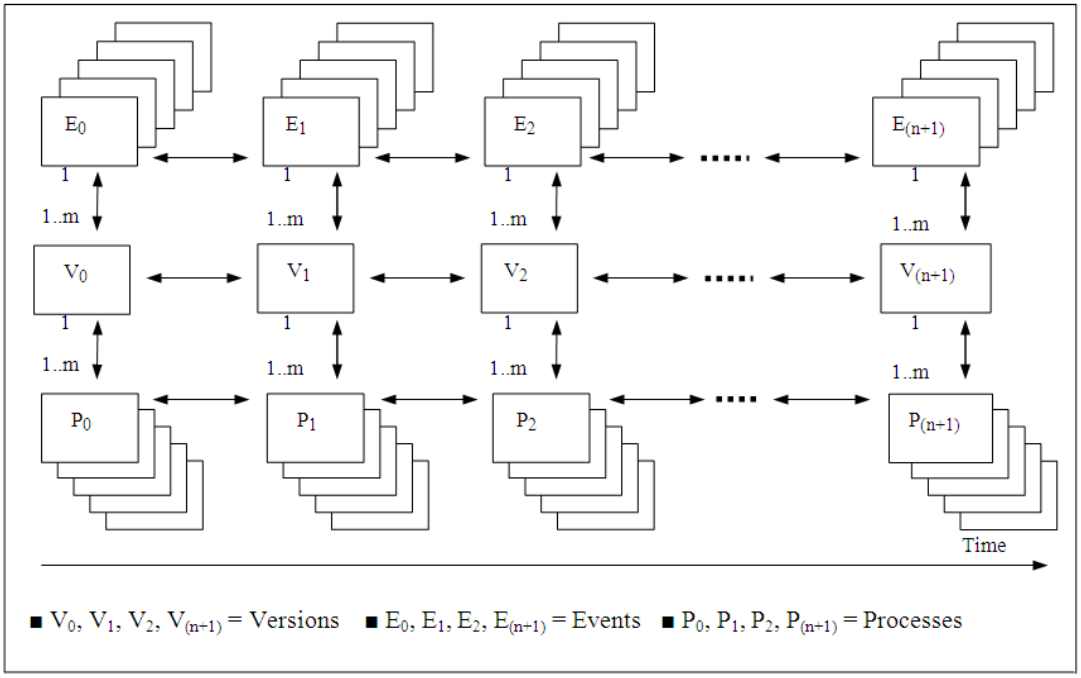

Fig. 4: Temporal links between versions, events and processes of an object

The process class has relationships with the spatial, thematic and temporal classes. The relationship between the process class and the spatial determines the actual affected section of the geographical object. Similarly, the relationship between the process class and the thematic class shows which themes were affected and the process-temporal class relationship indicates the actual time of the effect.

Moreover, there are no direct association relationships between the event class and the process class. There is, however, an initial relationship between the event class and the process through the version class. When an event occurs to a geographical object, dynamic attributes of the event are created. A process generates dynamic attributes when it is created. Subsequent temporal relationships between the event class and process class through the version class are set up by dynamic attributes of the events or processes[18]. Specific event(s) and process(es) is identified by the version of the geographical object. The relationship between the versions of a geographical object, the events and processes are shown in Fig. 4.

DESIGN OF THE MODEL

The visual modelling tool known as Unified Modelling Language (UML)[21] for object oriented modelling is used to describe the design of class hierarchies and their interacting relationships within the model. The UML describes the design of the versions and the relationships between the classes of the geographical objet.

As shown in Fig. 5, the attributes and behaviour of geographical object can be designated into independent classes that are inter-related. This technique allows specific changes of attributes and behaviour to be noted and altered over time without affecting the rest of attributes and behaviour of the geographical object. The UML is used to demonstrate the semantics of the versioning and temporal GIS by showing the versions of the geographical object with the events and processes related to the changes. The aggregated relationships between the version class and the spatial class, thematic class and the temporal class can be identified. The associated relationships between the version class and the event class and the process class can be determined. Each class is represented by a rectangle that has different compartments that show attributes and behaviour of each class (Fig. 5).

The rectangles have links to show the type of relationships between the classes. The class *Geographical_Object* represents a map or GIS output device, the *GIS_Object* class represents geographical objects such as road, building and the *Object_Versions* class represents the versions of the objects. The rest are the *Spatial*, *Temporal*, *Thematic*, *Events* and *Processes* classes.

The links represent aggregations and associations between the various classes, such as *Obj_type* representing the aggregated relationships between the map and the geographical object. The *Ob_versions* link represent the associated relationships between the object and the versions and *V_space* representing the aggregated relationships between the spatial class and the object class and the version class, others represent the corresponding links in a similar way.

The interaction between the classes which results from attributes and behaviour changes over time can be represented using the time sequence diagram (Fig. 6). The time sequence diagram shows the interaction between the attributes and behaviour changes such as spatial, thematic and temporal that results into versioning of the geographical object. Furthermore, the interaction between the geographical object, the events and the processes are represented. Also shown in Fig. 6 an event (flooding) which can produce a process such as the evacuation (movement) of people (attributes) from a town.

IMPLEMENTATION AND EVALUATION

The model was implemented using an object oriented database, Objectivity[22] and object oriented programming environment, visual C++ and running under Windows NT platform. This approach eliminates the need of mapping the model to an OODBS since the class structure used in the OOM, the OOPE and OODBS were consistent. As shown in Fig. 7, the classes (version, temporal, spatial, thematic, event and process) are defined in the application schema file known as the application Data Definition Language (DDL) schema. The DDL processor generate the schema header file and the schema source code which are linked with the application source code. In the application DDL schema and application source code files, the version class is represented by *GObject*, the spatial class by *Spatial*, the thematic class by *Thematic*, the temporal class by *Temporal*, the event class by *Event* and the process class by *Process.*

The relationships between the classes are established in the application DDL schema file using the object reference class function, *ooRef*. For example, when the first line of code below is inserted in the version class and the second in the spatial class, the relationships between the version class and the spatial class is created:

1. *ooRef(Spatial) current_spatdata;*
2. *ooRef(GObject) current_SpatObject;*

Objectivity/DB has the capabilities to represent the versioning approaches (i.e. linear, splitting and merging) demonstrated in the OOM. Linear changes are represented by linear versioning method using the *setVersStatus(oocLinearVers)* function. Changes involving splitting are represented by branching versioning technique using the *setVersStatus(oocBranchingVers)* function. Geographical phenomena involving merging are represented by the merging versioning approach using the *add_derivative* function. If the properties of the merged object are similar to the previous objects, the *add_derivedFrom* function is used. Versioning is established by invoking the *version(copy)* and *version(move)* function in the application DDL schema file. The *version(move)* function allow the attributes and behaviour of the previous version to moved to the current version and *version(copy)* function enables the copying of the properties of previous version to the current version.

Geographical objects are made to be persistence by storing the object in a federated database (divided into databases and containers). Persistence objects are identified using the object identifier (OID) which is unique within a federated database. Objectivity/DB uses the object handle class, ***Handle*****,** to access persistence objects automatically by the DDL process for every persistence class found in the schema header.

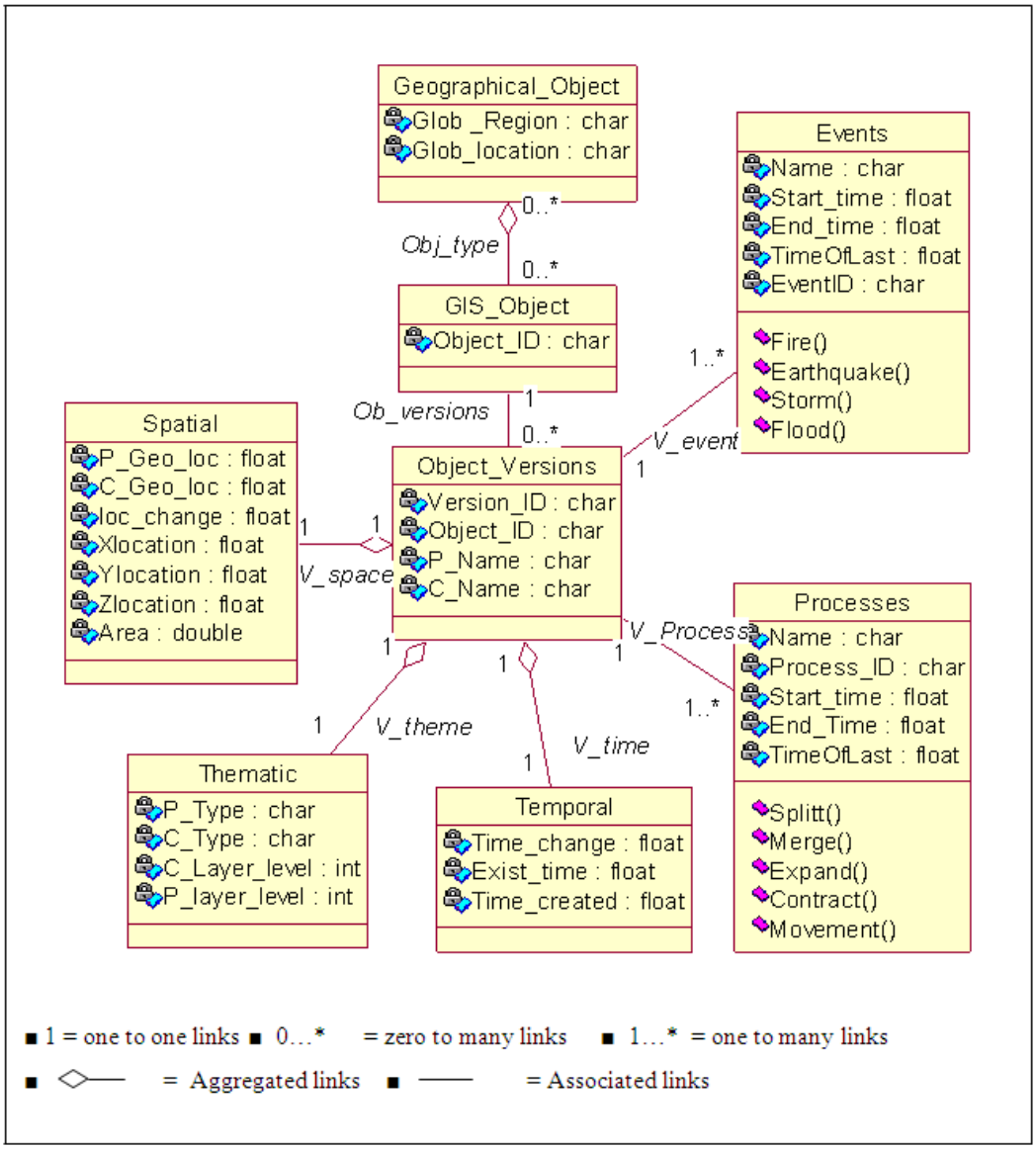

Fig. 5: UML design showing semantic of the versioning and temporal relationships

The persistence objects of the version class was represented by *GObjectH*, the spatial class by *SpatH*, the temporal class by *TempH*, the thematic class by *ThemH*, the process class by *ProcessH* and the event class by *EventH*. For example, the persistence objects of the version class and the spatial class are generated using the method below:

Handle (Spatial) SpatH; Handle (GObject) GObjectH; Aggregated relationships between the version class and the spatial class, the thematic class and the temporal class is established in the application source code using the code below:

GObject::GObject():
current_spatdata(newSpatial()),current_tempdata(new Temporal()),
current_themdata(new Thematic());

A dynamic function handles the temporal relationships between the versions, the events and the processes. The code below generated the relationships between the versions, events and processes. Each process is linked to the event through the version:

EventH=new(GObjectH) Event(TempEvent);
ProcessH=new(GObjectH);

Database query: The query technique of the system was classified into different components in order to identify the attribute and behaviour changes to the classes of the model. Since an object oriented approach has been used, the attributes and behaviour queries changes over time are directly related to geographical object. The query components of the system are independent of each other but are co-related.

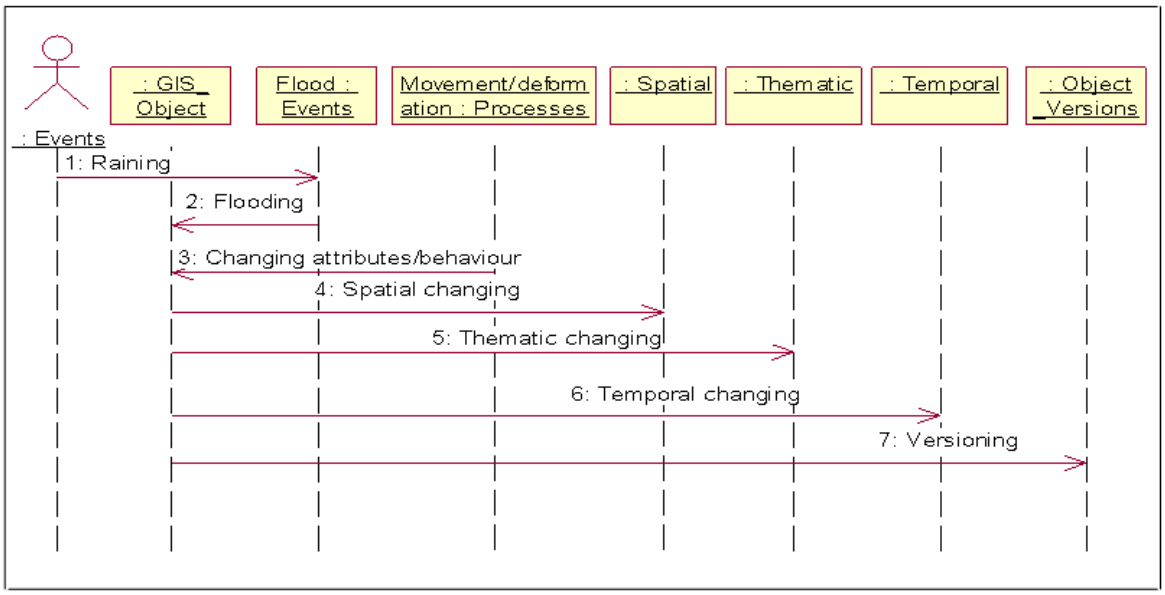

Fig. 6: Sequential diagram showing interaction of the changes of geographical object over time

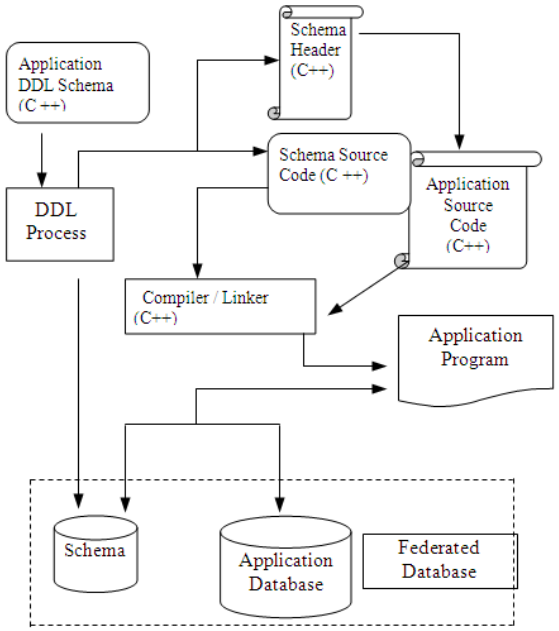

Fig. 7: General architecture of the OOM implementation

They are defined in relation to the spatial class, the thematic class, the temporal class, the event class and the process class. Temporal queries enable the time that the changes of the geographical object occurred, the time that event occurred and the time the process was noted to be determined. For example, “when were these houses changed to a supermarket?” This type of queries is based only on the time at which the changes occurred and it is implemented by iterating through the database to find the time at which the changes occurred. The sample in code 1 shows how such query was implemented.

The temporal queries capture the relationships between versions. For example, “when was this site changed from golf course to race course?” Spatial queries are based on changes such as area and location alterations or adjustments. For example, “which areas in the UK are affected by rainfall that causes heavy flooding?” Spatial topology relationship between versions will be determined using such spatial queries. For example, “which houses within ten kilometres of the river were affected by the flooding?”

```
ooStatus Map::Find_Temp_Infor()
{
  ……
  ……
 char input_N[20];
 cout<<"Enter required time"<<endl;
            gets(input_N);
 ///scan the container for the all the objects
            GObjectI.scan(contH);
 while (GObjectI.next())
            {
      …….
              if(!strcmp(GObjectI->current_tempdata->_created_
stamp,input_N))
                        {
                         GObjectI->GObject_Print();endl;cout<<
                        "Event"<<GObjectI->current_event->_   Event
                        Name<<endl;
                  cout<<"Process"<<GObjectI->current_
                        process->_ProcessName<<endl;
                         }
            }
 …….
}
```

Code 1: Sample of code for querying the database using time that the changes

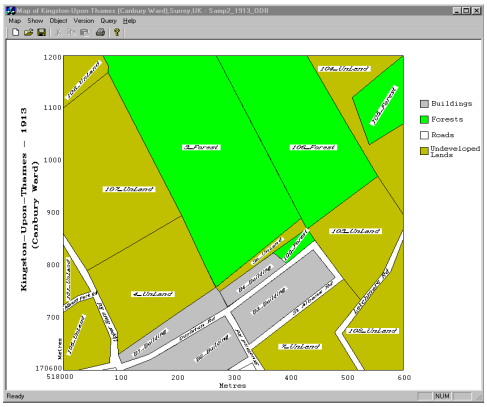

Fig. 8a: Canbury Ward Map 1913

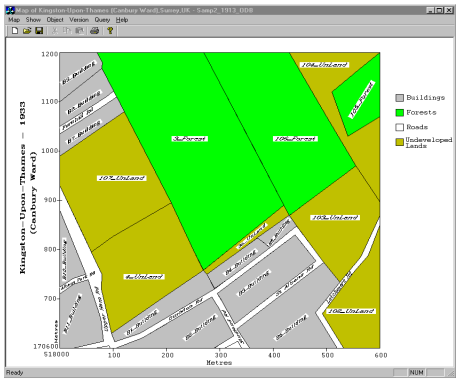

Fig. 8b: Canbury Ward Map 1933

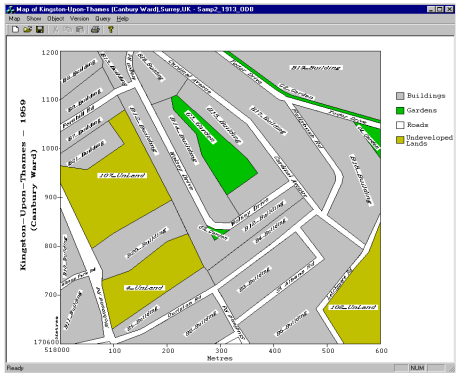

Fig. 8c: Canbury Ward Map 1959

Fig. 8d: Canbury Ward Map 1998

Spatial topology relationship between versions will also be determined using the spatial queries. For example, “which houses within ten kilometres of the river were affected by the flooding?” Thematic queries are related the changes of the theme (of the geographical object). For example, “which parks in this county have been replaced by commercial building?” The event queries provide the ability to determine the nature of an event that causes the changes and the relationships between the events over time. The process queries allow the examining of the effects of the changes and the relationships between subsequent actions.

As indicated earlier the relationships between the versions allow forward and backward movement. The previous version and the next version to the current version can be obtained by iterating either backward using the *getPrevVers* function *GObjectH.getPrevVers(GObjectH3);* or forward using the *getNextVers* function *GObjectH4.getNextVers(GObjectI);*.

All the objects in the database can be determined by scanning through the database using the iteration scanning function *GObjectI.scan(contH);*. Similarly, the current version can be obtained using the same approach.

Case study: The functionality of the integrated GIS system was demonstrated using a case study involving a data set from the Royal Borough of Kingston-Upon-Thames, Surrey, UK. This entailed tracking the attribute changes of geographical objects in the Canbury Ward during the period of 1913, 1933, 1959 and 1998. The data for testing of the system were obtained by scanning the historical paper maps to obtain raster images, followed by using ArcInfo Software (ESRI) to generate vector data. The vector data of the maps were stored in the database (Objectivity/DB) which are shown in Fig. 8, Fig. 8a represents 1913 map, Fig. 8b 1933 map, Fig. 8c 1959 map and Fig. 8d 1998 map.

The process of how a map (of 1913) is stored in the database (Objectivity/DB) is shown in Fig. 9. It starts with the object browser named **KingstonCW** consists of boxes labelled **Databases** and **Containers**, **Objects** and **object – “Durlston”**. The box labelled **Database** shows the name of the created database, **CW_DB**, in the federated database. The box labelled **Container** indicates the number of containers in the database, the box labelled **Object** shows the name of geographical objects in a container, for example, when the container labelled **Durlston** was highlighted it contains an object named **Durlston**. The box labelled **object – “Durlston”** show detailed information about the geographical object **Durlston Road**. Thus, the **Container** box shows the names of all the geographical objects in the 1913 historical map of Canbury Ward that were added to the federated database.

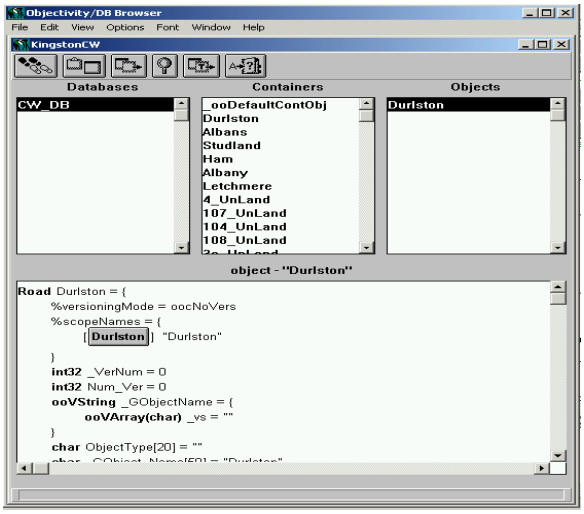

Fig. 9: Objectivity/DB object browser showing the contents of the federated

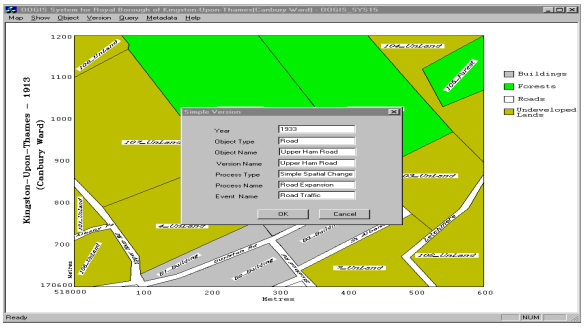

Fig. 10: Changing the attributes of the old Upper Ham Road and linking the event and process to the new version

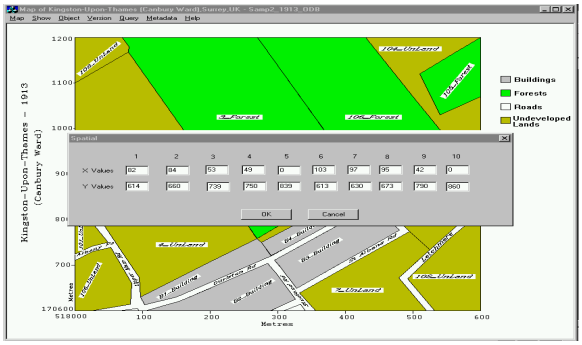

Fig. 11: Dialog box for updating the spatial attributes of a version of geographical object

Due to the fact that geographical objects experience different types of geographical phenomena, in this case study linear (simple), splitting and merging changes were tested. The forward and backward continuous tracking of the versions of geographical object over time were also tested.

Linear versioning: Testing of the changes of geographical objects that experienced linear versioning was based on spatial and thematic attribute changes of the objects. Comparing the historical map of Canbury Ward in 1913 (Fig. 8a) and that of 1933 (Fig. 8b), there was an expansion on the Upper Ham Road in 1933 due to traffic congestion in the area. This is represented in Fig. 10 where the old version of a geographical object (the Upper Ham Road) was linked to a newer version. The "Year" represents the year that changes were updated, the "Object Name" represents the name of the old version and the "Version Name" is the same since there was no changes in the name of the object.

The event and the process associated with the spatial attributes changes are also represented in Fig. 10. The "Process Name" represents the process, the "Process Type" represents the type of geographical phenomena (e.g. simple, split, merge or cyclic) and "Events Name" represents the event. The event linked to this geographical phenomenon is represented by the "Road traffic" and the process linked to the phenomenon is "Road Expansion".

Figure 11 shows a sample dialog box that was used to update the spatial attributes of the **Upper Ham Road**. The attribute changes (i.e. spatial) were updated to the new versions of the geographical objects. The spatial changes represent the Y (northing) and the X (easting) co-ordinates of the geographical objects.

The spatial attributes updated on the map by retrieving the values from the database as shown in Fig. 12. The updated spatial attributes are now representing the northing and easting co-ordinates of the new versions of the geographical object **Upper Ham Road**.

Split versioning: In 1959, sections of land parcel, 4_UnLand were used for developing a school building and are represented as B20_Building (Fig. 8c). The spatial changes of the geographical object, 4_UnLand in 1959 represents split versioning. The original geographical object, 4_UnLand has the same feature attributes as the new version 4_UnLand but different spatial attributes, however the new version 4_UnLand has different feature properties to the version B20_Building.

The dialog box for testing the split versioning is similar to the one for simple changes (Fig. 11) but it has an extra input for the name of the second version. The updated two versions 4_UnLand and B20_Building were stored in the same "container" of the federated database. Shown in Fig. 13 is the map of Canbury Ward in 1959 showing the updated geographical objects of 4_UnLand and B20_Building.

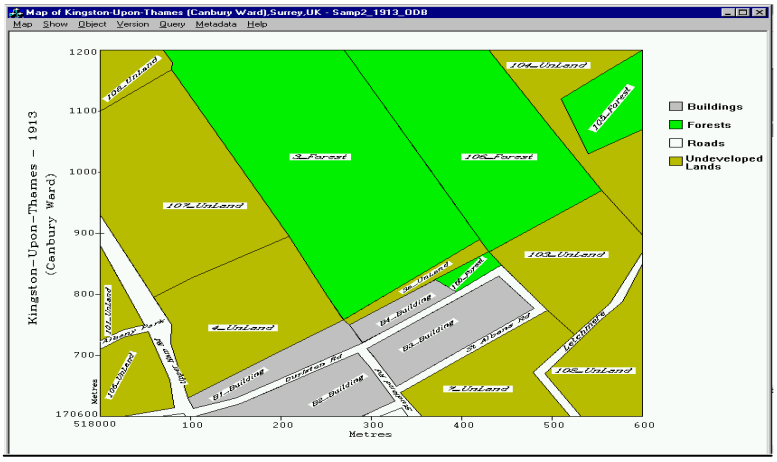

Fig. 12: Historical map of Canbury Ward in 1933 showing the new spatial attributes of the Upper Ham Road

Fig. 13: A map of Canbury Ward of 1959 showing only updated spatial changes of the versions created after the split versioning of the geographical object, 4_UnLand

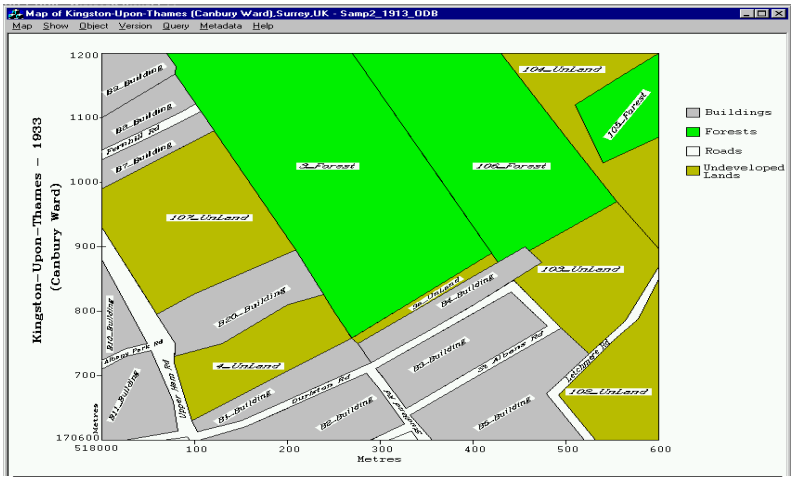

Fig. 14: A map of Canbury Ward of 1959 showing only updated spatial changes as a result of merging of geographical objects, 4_UnLand and 6_UnLand become the new object B4 Building

Merge versioning: The merge versioning technique of the system was tested by using the scenario, where the two geographical objects B4_Building and B6_Building merged in 1959 to become the new object

B4_Building as shown in Fig. 14. The new version B4_Building has the same features attributes as the previous two objects but its spatial attributes are equal to the combination of the two objects.

CONCLUSION

Different approaches to representing changes of geographical phenomena for analysing and tracking the evolution of objects have been discussed, many of which have not yet been successfully implemented by their exponents. The object oriented model, which is adopted in this research, tackles the limitations of many previous temporal GIS work and provides an integrated framework for the effective handling of continuous tracking of the evolution of geographical objects. The specific approach presented in the paper, produces a good unified object-oriented system comprising an object-oriented data model, object-oriented programming language and an object-oriented database. The proposed system eliminates the difficulties previously experienced in mapping spatial object-oriented data and relational models, which suffered from the mismatching of relationships between versions of the objects.

The proposed system constitutes good temporal modelling because the temporal attributes and behaviour of the versions are independent and have relationships patterns which enable tracking of changes. Moreover, the temporal attributes include temporal operators to promote the continuous analysis of patterns of change. The system also works well with both gradual and sudden changes because the attributes of events have temporal operators and versions have built-in relationships between them.

The GIS system proposed by the authors eliminates the need for large data storage capacities by recording only the changes in the spatial, temporal, thematic, event and process classes. The results have also indicated that continuous tracking of the patterns of change of geographical phenomenon can be achieved effectively. The development of an improved graphical user interface together with interfacing with some existing geographical information systems will be tackled in future work.

REFERENCES

1. Sui, D.Z., 1998. GIS-based urban modeling: practices, problems and prospects. Intl. J. Geograph. Inform. Syst., 8: 7-24.
2. Adamu A.S. Khaddaj and M. Morad, 2001. A framework for tracking the evolution of objects in GIS. In D.B. Kidner and G. Higgs G (Eds) GIS Research–UK Cardiff University, pp: 304-310.
3. Langran, G., 1993. Manipulation and analysis of temporal GIS. In Proc. of the Canadian Conf. on GIS, Canada, pp: 869-879.
4. Yourdon, E., 1994. Object-Oriented System Design: An Integrated Approach. Yourdon Press.
5. Wachowicz, M., 1999. Object-oriented design for temporal GIS. Taylor & Francis, London.
6. Yuan, M., 1997 An intelligent GIS to support spatiotemporal modeling in Hydrolog. http://ncgia.ucgia.edu/conf/SANTA_FE_CD-ROM/ sf_paper / yuan_may / may.html.
7. Milne, P., S. Milton, J.L. Smith, Geographical object-oriented database- A case study. Intl. J. Geograph. Inform. Syst., 7: 39-55.
8. Loomis, M.E.S., Hitting the Relational Wall. J. Object-Oriented Programming, pp: 56-59.
9. Worboys, M., 1994. Object-oriented approaches to geo-referenced information. Intl. J. Geograph. Inform. Syst., 6: 353-399.
10. Wachowicz, M. and R. Healey, 1994. Towards temporality in GIS. In Worboys M. F. Innovation in GIS I, Taylor & Francis, 1: 105- 115.
11. Hornsby, K. and M. Egenhofer, 2000. Identity-based change: A foundation for spatio- temporal knowledge representation. Intl. J. Geograph. Inform. Syst., 14: 207-224.
12. Peuquet, D.L. and Qian, 1996. An integrated database design for temporal GIS. In M. Kraak and M. Molenaar (eds). Spatio-temporal I, Spatio-Temporal Data I in Advances in GIS Research.
13. Frank, A., 1994. Qualitative temporal reasoning in GIS-ordered time scales. In Sixth Intl. Sym. on Spatial Data Handling Edinburgh Scotland International Geographical Union, pp: 410-30.
14. Peuquet, D. and N. Duan, 1995. An event-based spatio-temporal data model (ESTDM) for temporal analysis of geographical data. Intl. J. Geograph. Inform. Syst., 9: 7-24.
15. Claramunt, C.P., C.S. Spaccapietra and M. Theriault., 1999 Database Modelling for Environmental and Land Use Changes in Stillwell J. Geertman S. and S. Openshaw (Eds) Geographical Information and Planning, pp: 182-202.
17. Peuquet, D., 1998. Time in GIS and geographical databases. In Longley P. Goodchild M., Maguire D. and Rhind D (Eds). Geographical Information Systems: Principles and Technical Issues, Vol. 1 chap. 8.
18. Owen, P.K., 1993. Dynamic functions triggers in an on-line topology environment. In European Conf. on Geograph. Inform. Syst. (EGIS), 2: 1249-1255.
16. Claramunt, C. and M. Theriault, 1996. Towards semantics for modeling spatio-temporal processes within GIS. 7th Intl. Sym. on Spatial Data Handling SDH'96, pp: 12-16.
19. Dadam, P., V. Lum and H.D. Werner, 1984. Integrating of time versions into relational database systems. In Proc. Conf. on Very Large Database, pp: 509-522.
20. Allen, J.F., 1984. Towards a general theory of action and time artificial intelligence, 23: 123-154.
21. Quantrani, T., 2000. Visual Modelling with Rational Rose 2000 and UML. Addison-Wesley.
22. Objectivity/DB 2000. Complete handbook for objectivity/C++ Instruction Manual. Objectivity Press.