IADIS International Journal on Computer Science and Information Systems
Vol. 8, No. 1, pp. 79-96
ISSN: 1646-3692

VISUALIZATION OF GEO-REFERENCED ENTITY: AN ASPECT-ORIENTED PATTERN

Armanda Rodrigues. *CITI/FCT, Universidade Nova de Lisboa, Quinta da Torre, 2829 -516 CAPARICA, PORTUGAL.*

Sara Silva,*CITI/FCT, Universidade Nova de Lisboa, Quinta da Torre, 2829 -516 CAPARICA, PORTUGAL.*

João Araújo. *CITI/FCT, Universidade Nova de Lisboa, Quinta da Torre, 2829 -516 CAPARICA, PORTUGAL.*

ABSTRACT

Web Geographic Information Systems (GIS) are systems composed by software, hardware, spatial data and computing operations, which aim to collect, model, store, share, retrieve, manipulate and display geographically referenced data. The development of online geospatial applications is currently on the rise, but this type of application often involves dealing with concerns (i.e., properties) which are inherently volatile, implying a considerable effort for system evolution. Nevertheless, geospatial concerns (e.g., temporarily blocked streets), although changeable, are reusable. However, lack of modularization in software artifacts (including system's models) can compromise reusability. In this context, the use of requirements analysis patterns, enriched with aspect-oriented modeling techniques, can support reusability and improve modularity. In this paper, we introduce requirements analysis patterns for geospatial concerns, to facilitate modularity in GIS Web applications. These patterns are generated from the domain analysis of Web GIS applications and described using a template which is supported by a comprehensive tool, enabling the completion of specific geospatial patterns.

KEYWORDS

Web GIS; Analysis Patterns; Spatial Concerns; Aspect-Oriented Modeling.

1. INTRODUCTION

According to Dragicevic (2004), a Geographic Information System (GIS) is a computer system that supports the use and handling of geospatial data. GIS are mainly information systems composed of software, hardware, spatial data and computing operations, which aim to collect, model, store, share, retrieve, manipulate and display geographically referenced data. Web GIS involves the online availability of Geospatial data, with the associated tools.

The development and availability of online geospatial applications is currently on the rise. This type of application often involves the temporary availability of spatial concerns (i.e., properties), inherently volatile, although recurring and, therefore, reusable (e.g., temporarily blocked streets). This implies a considerable need for maintenance, not only of the respective information structures, but also of its dynamic behavior. Also, the lack of modularization can compromise flexibility and lead to reuse problems. Moreover, maintenance must be undertaken with full knowledge of the application domain and using proper technical maintenance support techniques.

Based on knowledge of the application domain, resulting from preliminary domain analysis, the identification of reusable (in our case, spatial) concerns (e.g., map adjustment according to temporal conditions or geographic interfaces) will facilitate Web GIS development. To help with this task at early stages of software development, requirements analysis patterns may be applied to spatial concerns. Analysis patterns are reusable specifications, used at early stages of the development process. A pattern is a description of a recurrent problem through a solution model, which can be reused in different contexts (Gamma *et al*, 1995; Hamza and Fayad, 2004). The reuse of these patterns and their instantiation in the analysis models of a particular Web GIS application will speed the development process.

We propose an approach to improve modularization when modeling Web GIS during the requirements specification. The aim of this work is thus to promote requirements modularity and reusability and hence the evolution of Web GIS applications. Nevertheless, the reuse of spatial concerns depends on the availability of appropriate modularization and composition mechanisms. It is important to identify, not only spatial concerns, but also other concerns which are related to these. Moreover, the transverse quality of these concerns must be taken into account, as relevant spatial concerns may crosscut various parts of a particular application.

Aspect-Oriented Software Development (AOSD) is characterized by allowing the identification, modularization and composition of crosscutting properties (or concerns) (Filman *et al*, 2005). One property is said to be crosscutting if it is tangled with another property in a single module or if it is scattered in several system modules. Aspect-orientation is a software reuse paradigm and perfectly suits the specification of patterns' models, as it provides efficient mechanisms to reuse and compose pattern's models to a specific application. In this work we adopt MATA (Modeling Aspects Using a Transformation Approach) (Whittle *et al*, 2009), a technique for modeling and composition of patterns based on graph transformations.

In summary, the aim of this work is to create an aspect-oriented requirements analysis approach to model volatile but reusable concerns in Web GIS, specifically geospatial properties, based on patterns. To efficiently support the development of any type of application, it is more productive to have its domain analysis available, and a catalogue of

associated analysis patterns. Geospatial applications are no exception. It is also important to define these patterns using an appropriate template for geospatial applications, whose solution models are specified using aspect-oriented principles, enabling the systematic reuse of spatial properties. This paper complements previous work where a pattern to locate an entity was defined (Silva *et al.* 2012).

The remaining of this paper is organized as follows. Section 2 describes spatial concerns and the MATA approach. Section 3 presents the proposed requirements analysis pattern template. Section 4 applies the pattern to a spatial concern. Section 5 describes the tool support and section 6 depicts some related work. Section 7 draws some conclusions and points directions for future work.

2. BACKGROUND

2.1 Spatial Concerns

The use of GIS implies the handling of large volumes of data which are visually integrated in a spatial framework and which, at the same time, need the availability of efficient and adapted data manipulation operations. The highly dimensional operation of GIS data involves the development of complex applications, where relevant concerns may crosscut various parts of a particular application.

The availability of map APIs, enabling the development of geospatial components for existing web services (e.g., Google, Flickr, Facebook, etc.) along with the popularization of the use of Global Positioning Services (GPS), has led to the growing numbers of Location-Based Applications on the Internet and Mobile Internet. We will name this type of applications, Web GIS applications. They have taken the GIS concept, which was previously mainly used by natural sciences professionals and researchers, to the widespread theatre of the Internet and Mobile Devices, providing location-based adaptations of existing popular services. This means that searching for a favorite type of restaurant can now be made around the area where a smart phone is located. Also, you can quickly discover which is the nearest cinema presenting the film that you want to watch tonight. The web context adds an additional difficulty to the development of GIS applications: requirements volatility (Sommerville, 2010). Not only data and availability are constantly changing but requirements evolve also. These requirements are directly related to the spatial needs behind Web GIS users and they can be identified by the use of a spatial concerns catalogue, in association with the conception of an approach for modeling spatial concerns using aspects in Web GIS applications (Oliveira *et al*, 2010). This catalogue is currently under development (Oliveira *et al*, 2010) and the results of the work presented in this paper will be added to it. Mainly, typical spatial concerns can be separated in five categories, described as follows:

- Spatial Business Objects: To enhance applications by adding a spatial mapping and representation to business objects (e.g., a bus service management system can be improved by providing real-time bus locations);
- Rich Spatial Data: Enriching a geographic object with additional (not geo-referenced) information (e.g. adding a video to a specific location on a map);

- Spatially-Constrained Behavior: Change or modify the behavior of an object according to its actual location (e.g. pricing and taxing processes may change with objects' locations);
- Map adjustments: To extend or restrict the available spatial information according to the application's constraints (e.g. certain parts of a map may be, according to temporal or permanent restrictions, unavailable or useless for specific operations).
- Geographic Interfaces: To modify (or upgrade) the user interface of geographic objects. Though this is not strictly a spatial concern, it is clear that the previously described concerns may introduce changes in the application's user interface, specifically in the representation of geographic objects (e.g., to introduce a particular symbolization to make the user aware that a road cannot be used during a particular period of the day).

By addressing these types of (spatial) concerns we aim to identify the situations in which they may be present in web applications and to develop an approach to handle them early on in the development process, specifically during requirements specification.

2.2 Aspect-oriented Modeling

The MATA aspect-oriented modeling approach (Whittle *et al*, 2009) is based on UML, allowing aspects composition using, for example, class diagrams, sequence diagrams and state diagrams. Here we focus on MATA to model aspectual classes by using and adapting class diagrams.

Variables in MATA are prefixed by a vertical bar "|", meaning that "|X" will match any model element with the same type of X. After specifying both kinds of models, base and aspectual, a pattern matching is made between them. This means that the MATA tool tries to establish a connection between elements of each model, always respecting the composition rules defined in the aspectual model. The resulting composed model includes the elements of both models, according to the rules defined. MATA allows more composition combinations than other existing aspect-oriented modeling tools.

3. ASPECTUAL REQUIREMENTS ANALYSIS PATTERN TEMPLATE

The template used to define requirements analysis patterns – Table 1 – is based on the one proposed in Pantoquilho *et al* (2003), but adopting MATA aspectual notation for the modeling part. The motivation to define and use a new pattern template instead of adopting an existing one (e.g. Fowler, 1997; Gamma *et al*, 1995) is that neither do the current templates support composition mechanisms nor do they provide more detailed models such as variability models (Kang *et al*, 1990) or non-functional requirements (Sommerville, 2010).

Table 1. Analysis Pattern Template

Field			Description
Name			Pattern name identifier.
Problem			Describes the problem that the pattern intends to address.
Context			Describes the environment in which the problem and solution apply.
Requirements	Functional		List of functional requirements.
	Non-Functional		List of non-functional requirements.
	Dependencies		Identification of dependencies between the functional and non-functional requirements.
	Actors		Identifies pattern actors.
Modeling	Structure	Feature Model	Shows optional and mandatory features of the pattern.
		Class Diagram	Shows the pattern's classes and relationships
	Behavior	Sequence Diagram	Shows the dynamic behavior of the pattern.
Consequences			Advantages and disadvantages of the pattern's application.
Events List			Identifies the events that trigger the pattern.
Examples	Features Diagram		Examples that illustrate the context of the pattern, as it is applied and any necessary amendments to the initial context.
	Class Diagram		
	Sequence Diagram		
List of related patterns.			List of already defined patterns related to the proposed one.

The proposed template begins with the name of the pattern, followed by the description of the problem which it seeks to solve, the problem context and applicability of this pattern. The functional and non-functional requirements of this pattern are also listed as well as the dependencies among these requirements. Then, the pattern's models are provided: a feature model (Kang *et al*, 1990) (to represent variability), a class diagram (for the domain classes) and a sequence diagram (to specify behavior). Finally, examples are presented where this pattern could be applied, the consequences of its application and the patterns that relate to it.

For the creation of patterns, an analysis of Web GIS applications to identify spatial properties that would be reusable must be realized beforehand. Afterwards, analysis patterns are identified and then described using the template presented in Table 1. Thus, the description of a pattern requires a meaningful name. Then, it is necessary to describe what problem the proposed pattern seeks to solve and explain the context in which it commonly appears namely what kind of situations it is relevant to. After introducing the purpose of the pattern, its requirements must be described. This description consists of a list of functional and non-functional requirements, followed by the identification of dependencies between them and the actors that will take action on the problem.

After specifying the requirements, the next step is to build the structural and behavioral models. The structural modeling consists of describing: which features should be present in the solution, represented in a feature diagram highlighting the variability of the pattern; and

which classes will be needed to implement the solution, represented by a MATA (or aspectual) class diagram, which will identify concrete and variable classes and their relationships. Variable classes are the ones that need to be instantiated when composing with base classes of an application. Therefore, the aspectual class diagram will describe the pattern structure, and the examples will show the composition of the base structure with the aspectual structure.

The following step is behavioral modeling, describing the behavior that the solution must have, or what activities will be required to solve the problem. The behavioral modeling will be done through a sequence diagram, also supported by AOSD methodologies, to facilitate the modularization. More specifically, we used scenarios and aspectual composition, realized through the mechanisms offered by MATA (Whittle *et al*, 2009). Thus, the sequence diagram will describe the aspect that represents the pattern behavior, and the examples will show the composition of the base scenario with the aspectual scenario, as illustrated in the previous section. Once the modeling is completed, we describe the consequences of the pattern's application, particularly its strengths and limitations.

Some events that would likely trigger the pattern are also specified. Some examples of applications of the pattern should also be provided. To describe these examples, features, class and sequence diagrams are configured for a particular application. The examples will include diagrams for the base and composed scenarios.

After the entire pattern is described, an analysis is made in order to identify possible relationships between this pattern and other patterns, which results in a list of related patterns.

At the end of this process, a detailed description of the pattern is provided, which can be reused in other applications.

4. PATTERN DEFINITION

One of the aims of this work is to identify spatial properties in Web GIS applications, described using analysis patterns. The *Show a Geo-referenced Entity* pattern identified in this work, and it is presented below.

Name: *Show a Geo-referenced Entity*

Problem: For users to benefit from a Geo-referenced Web application, geo-referenced entities must contemplate a visual representation. This means that some cartographic representation of the application's data set must exist, and each geo-referenced object must have a visual representation with possibly spatial behavior associated with it. Therefore, a change associated with the object's data may involve a change in its visual representation.

Context: This can be applied to the handling of geospatial objects with, initially, no visual representation but may also apply to objects which have experienced changes in their state, thus being subjected to change in their visual representation also. For example, a street that is closed for works may be drawn differently from other open streets. This change maybe volatile, time constrained. Volatility may also be associated with the objects location, itself. The assumptions taken are that the applications' entities are either static (do not move in space, e.g.: a garage) or mobile (may move during the execution of the application, e.g.: a bus

or a person). A static entity is associated with a location, while a mobile entity may be associated with several locations, during the execution of the application. It is also assumed that a mobile entity holds a main location (which can be home, an office, a garage) but, as it moves, it may become associated with a secondary location. This secondary location may be obtained from the analysis of the entity's schedule and requested from static entities, referenced in the schedule. This means that a bus may be located at a bus stop, which is a static location, or on route to the next stop and, at this time, its location is the street it is traversing, which is also a static location. As the location of the object changes, its state may also change and this may involve a change in the way the object if presented to the user, in the map.

Requirements:

Functional

1. Get geo-referenced entity;
2. Check if the entity is static or mobile:
   2.1. If static, obtain the geographic reference data;
   2.2. If mobile obtain system time and entity's schedule:
      2.2.1. Obtain the details of the entity's location;
3. Get the graphical representation of the entity (may be an XML representation);
4. Check if there are any restrictions / information to add to the entity
   4.1. If there are, add the restrictions / information relating to the representation of the entity;
5. Get map;
6. Overlap the graphical representation of the entity to the map;

Non-Functional

1. Correctness:
   1.1. In obtaining the entity;
   1.2. At checking whether the entity is static or mobile;
   1.3. In obtaining the geographic reference of the static entity;
   1.4. In obtaining the system time;
   1.5. In obtaining the schedule associated with the entity;
   1.6. At checking if the entity is in a secondary location;
   1.7. In obtaining the main and secondary locations;
   1.8. In obtaining the graphical representation of the entity;
   1.9. In verifying and obtaining restrictions;
   1.10. In overlaying geo-referenced objects on the map;
2. Response Time:
   2.1. when checking whether the entity is static or mobile;
   2.2. when obtaining the geographic reference of the static entity;
   2.3. when obtaining the system time;
   2.4. when obtaining the schedule associated with the entity;
   2.5. when obtaining the main and secondary locations;
   2.6. when obtaining restrictions;
   2.7. when obtaining the map;
   2.8. when overlaying the object on the map.

3. Precision:
   3.1. when obtaining the geographic reference of the static entity;
   3.2. when obtaining the system time;
   3.3. when obtaining the schedule associated with the entity;
   3.4. when obtaining the object's main and secondary location;
   3.5. when obtaining the object's graphical representation;
   3.6. when obtaining restrictions;
   3.7. when overlaying the object on the map.

Dependencies

Table 2. Dependencies between requirements

Functional Requirement	Depends on	Non-Functional Requirement
RF 1	→	RNF 1.1
RF 2	→	RNF 1.2
		RNF 2.1
RF 2.1	→	RNF 1.3
		RNF 2.2
		RNF 3.1
RF 2.2	→	RNF 1.4
		RNF 1.5
		RNF 2.3
		RNF 2.4
		RNF 3.2
		RNF 3.3
RF 2.2.1	→	RNF 1.6
		RNF 1.7
		RNF 2.5
		RNF 3.4
RF 3	→	RNF 1.8
		RNF 3.5
RF 4, 4.1	→	RNF 1.9
		RNF 2.6
		RNF 3.6
RF 5	→	RNF 2.7
RF 6	→	RNF 1.10
		RNF 2.8
		RNF 3.7

Actors: User; Entities; System/Application;

Modeling: *Structural*

Feature Model:

Fig. 1 illustrates a feature model to *Show a Geo-referenced Entity*. This model shows the variable and mandatory features associate to the pattern.

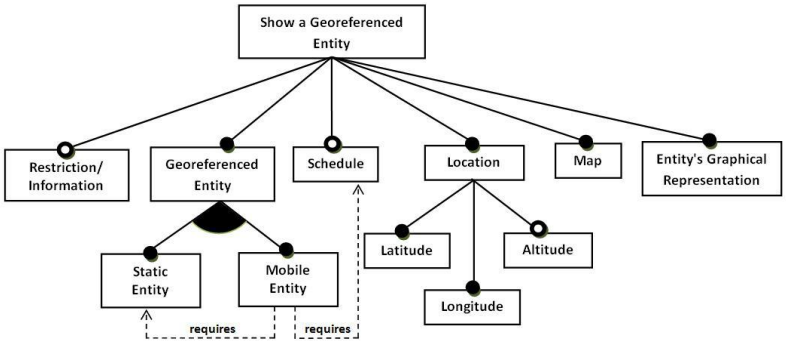

Figure 1. Feature Model for *Show Geo-referenced Entity*

There must be a Geo-referenced Entity which can be a Static Entity or Mobile Entity and cannot be both at the same time. Location is also compulsory, and this has a Longitude and Latitude and may or may not have an Altitude. Optionally, we can still have a Schedule. When we have a Mobile Entity, it must be associated with, at least, one Static Entity and a Schedule. Optionally, we can have Restriction or Information. A map and the entity's graphical representation are also compulsory. Besides specifying the variability of the pattern, this model can be used to help defining the class diagram as some features can be mapped to domain classes.

Class Diagram:

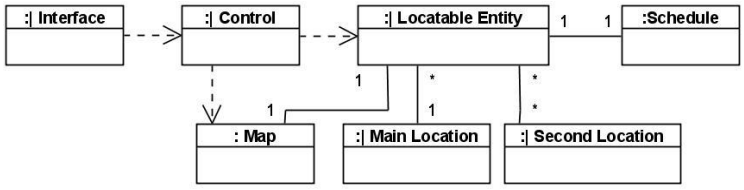

Figure 2. Class Diagram for Show Geo-referenced Entity

In the diagram in Fig. 2 we have seven classes: |*Interface*, |*Control*, |*Locatable Entity*, *Schedule*, |*MainLocation*, |*Secondary Location* and *Map*. The class |*Interface* is the intermediary between the user and |*Control*, and the latter (the |*Control* class) is responsible for checking and request the necessary data. The |*Control* class communicates with the |*Locatable Entity* and *Map* classes. The |*Locatable Entity* has a single *Schedule*, one |*Main Location* and can have several |*Secondary Location* and may be in several *Maps*. One *Schedule* only belongs to one *Locatable Entity*. |*Main Location*, as well as |*Secondary Location* can be connected to several |*Locatable Entity*, and a *Map* may have the representation of several |*Locatable Entity*. Note that we defined those classes as variables (except Schedule and Map).

Sequence Diagram:

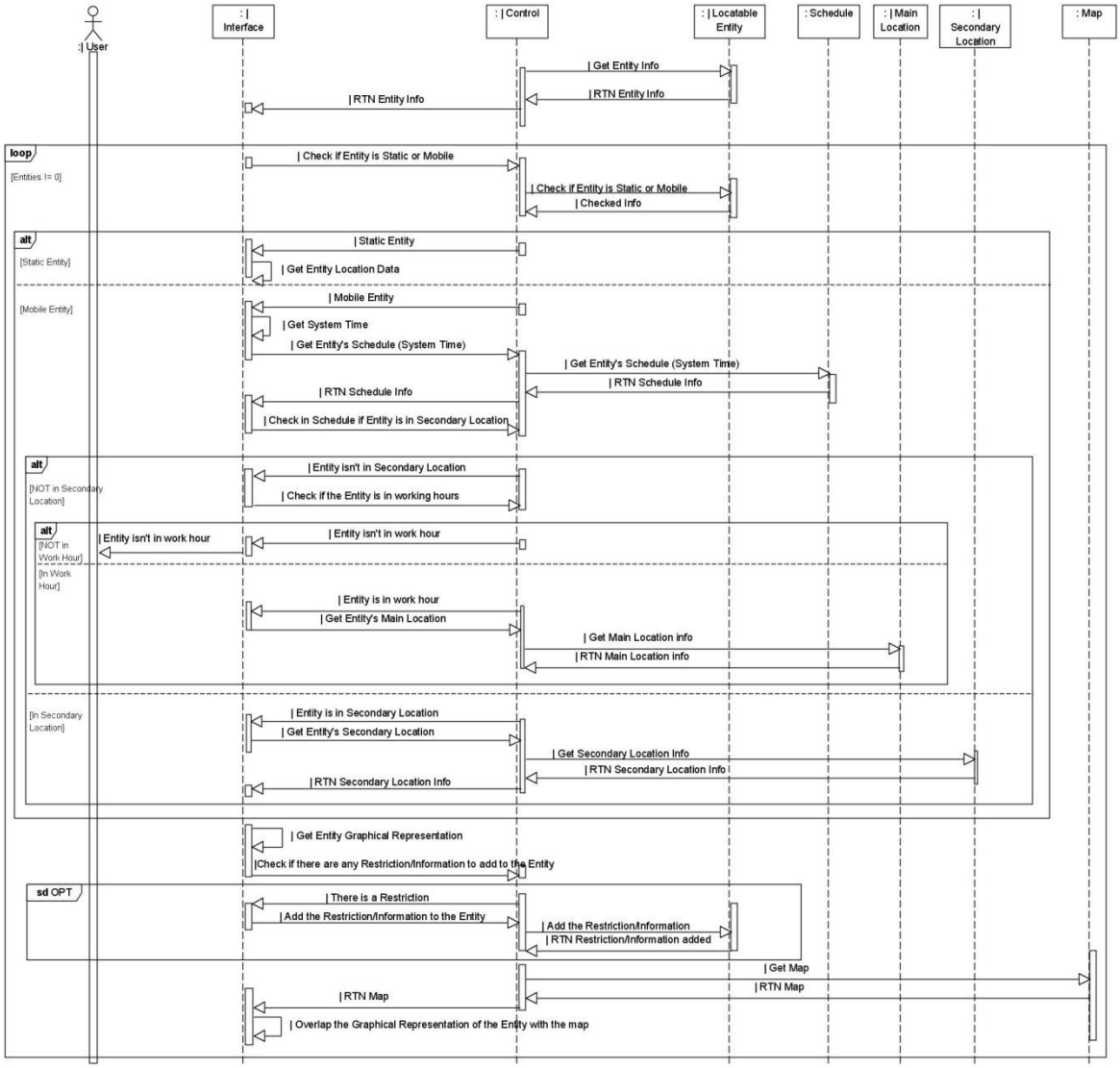

Figure 3. Sequence Diagram for Show Geo-referenced Entity

Figure 3 shows the Sequence diagram for the aspectual scenario. It describes the behaviour for presenting the cartographic representation of a geo-referenced entity to the user. This initiates when |*Control* requests information on |*Locatable Entity* and verifyies its mobility status (static or mobile). If static, |*Control* will inform |*Interface*, which will obtain its location. Otherwise, |*Interface* will also obtain the system time and will request, from |*Control* l, the |*Locatable Entity*'s Schedule. |*Control* will contact the :*Schedule* Entity for this

information and based on the entity's schedule, the location is determined. If the location can be determined from the schedule, the entity will be associated a secondary location. If the entity's location cannot be determined from the schedule, the location will be associated with the entity's main location, if the system time is currently working hours. If not, |*Control* will inform |*Interface* that the location of the |*Locatable Entity* cannot currently be determined. In the case when the location can be determined, |*Control* will solicit the location from the specific location entity associated with the |*Locatable Entity* (either Main or Secondary). This information will then be forwarded to |*Interface*, which will obtain the Entity's Graphical Representation. |*Interface* will also request any Restriction/Information to associate to |*Locatable Entity*, from |*Control*. If this information exists, |*Interface* will require its introduction by |*Control* in |*Locatable Entity*. Finally, |*Control* will obtain the *Map* and send it to |*Interface*, where the Entity Graphical Representation will be overlaid.

Consequences: The application of this pattern will enable the spatial location of an application's entity to be presented visually, for example, in a map.

Events List: Possible events are:

1. Fire breaks out in a building and you must find all the occupants therein to ensure that everyone leaves the building;
2. A bus company needs to inform its users, in real time, of the location of a given bus, showing it visually on a map;
3. A company that provides technical services needs a system that enables the real time visualization of the spatial locations of its technicians, to best manage the work.

Example: Here we show the actual application of the pattern models described above. This pattern can be applied to static entities, i.e. entities that have always the same location and in this case, this is simply stored in a database. Moreover, the pattern can also be applied to mobile entities, i.e. entities that change location according to certain characteristics, such as an individual or a vehicle. As an example, let us consider the application of this pattern to the CLIP[1] system, a real university information system, which provides course and schedule information of all students, teachers and other employees at Universidade Nova de Lisboa. In this system, individuals and classes are considered mobile entities, while rooms and offices are static. Let us look at a locating mobile entity example: an emergency happens and we need to convene a faculty staff meeting. Faculty members' locations must be resolved, and the Lecturer entity must be visually represented– that is the application of the pattern.

Feature Diagram: Fig. 4 has the necessary characteristics to show a lecturer – Show Lecturer Entity. As you can see, you need an object of class Map, Lecturer, its Graphical Representation, its Schedule and its Location, and the latter must be composed of a Latitude and Longitude, and may also include an Altitude. The Lecturer may also have an Office and/or a Room.

[1]http://clip.unl.pt

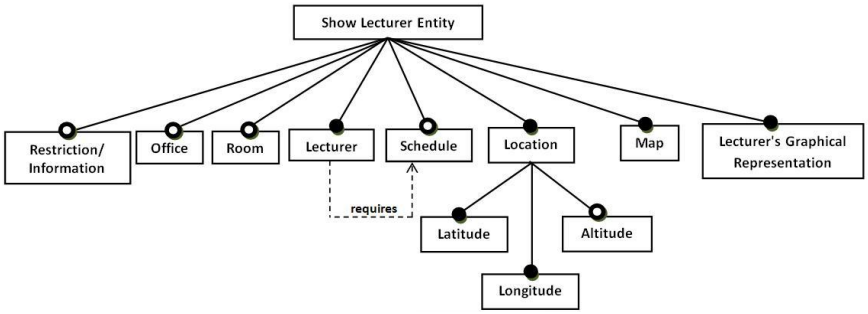

Figure 4. Feature Diagram for Show Geo-referenced Entity

Class Diagram: Fig. 5 illustrates the class diagram containing only the classes related to the objects that appear in the base scenario; this scenario corresponds to the representation in a map of lecturers called for a meeting. Thus, it will need *CLIP Interface*, *CLIP Control*, *Lecturer* and *Meeting*classes.

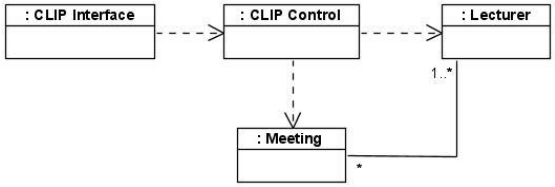

Figure 5. Class Diagram for Lecturer Base Scenario

Fig. 6 shows the class diagram for the base scenario composed with the aspect. So we have the aspect's classes instantiated with the base classes and composed with them. We thus instantiated aspect classes |*Interface* with *CLIP Interface*, |*Control* with *CLIP Control*, |*Locatable Entity* with *Lecturer*, |*Main Location* with *Office* and |*Secondary Location* with *Class Room*. The classes *Schedule* and *Map* don't need to be instantiated and from base scenario added the *Meeting* class.

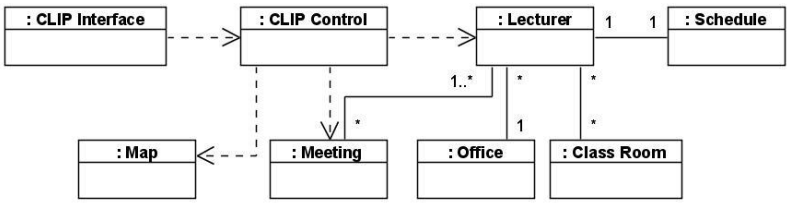

Figure 6. Class Diagram for Lecturer Composed Scenario

Sequence Diagram: Fig. 7 shows the sequence diagram for the base scenario in the example. The scenario is initiated with a request of contact of lecturers to attend a meeting on the user's part. *CLIP Interface* solicits information on *Meeting* from *CLIP Control*, and *CLIP Control* requests this information from *Meeting*. The returned information will include the list of summoned lecturers. Based on these data, *CLIP Control* will request the necessary

information from the *Lecturer* Entity which will the be forwarded to *CLIP Interface*, responsible for presenting it to the user.

The sequence diagram for the composed scenario is presented in Fig. 8, which shows the composition of the base behviour with the aspectual scenario. Thus, the Aspectual entities are instanciated by the ones taken from the example and the base entities are added to this. The instanciated aspectual entities are: |*Interface* with *CLIP Interface*, |*Control* with *CLIP Control*, |*Locatable Entity* with *Lecturer*, |*Main Location* with *Office* e |*Secondary Location* with *Class Room*. *Map* e *Schedule* do not need to be instanciated and the Meeting class is added from the base scenario. This diagram shows the base scenario until the moment when the lecturer information is provided to *CLIP Control*, time at which the aspectual scenario takes over (see fig.3). Once the end of the aspectual behaviour is reached, the base scenario resumes.

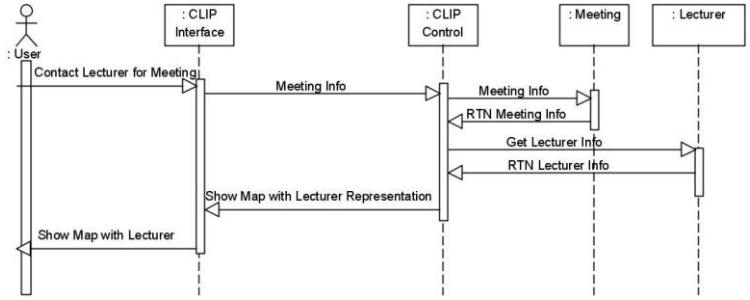

Figure 7. Sequence Diagram for Show Lecturer Base Scenario

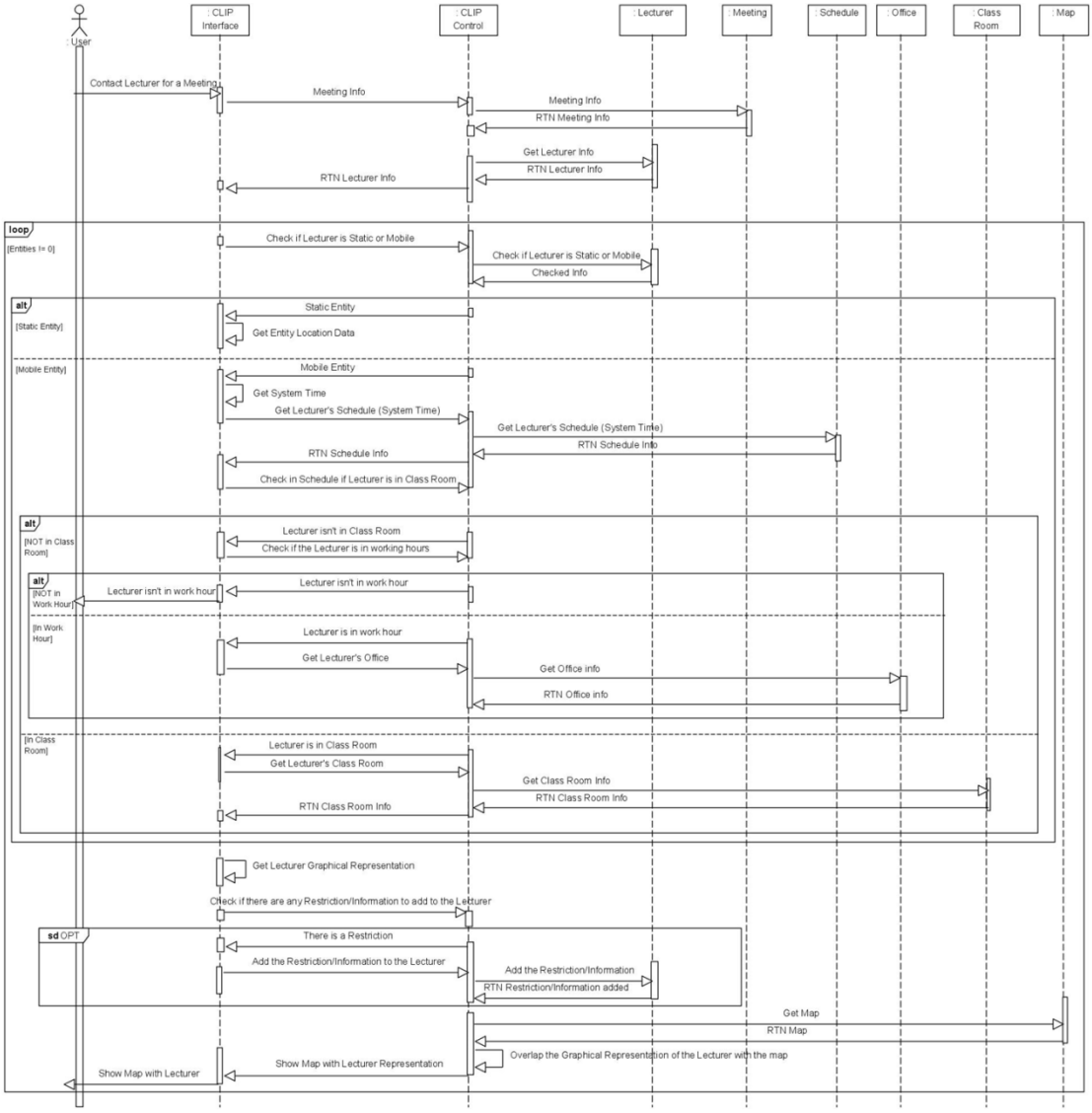

Figure 8. Sequence Diagram for Show Lecturer Composed Scenario

Related Patterns

- Add temporal availability for spatial entities;
- Adjust the state of entities (Add/Remove all or part of spatial entities).

5. PATTERN TOOL

The Pattern Tool tool was created to support this template, allowing the creation of patterns using the Eclipse environment. The generic view of the tool can be seen in Fig.9.

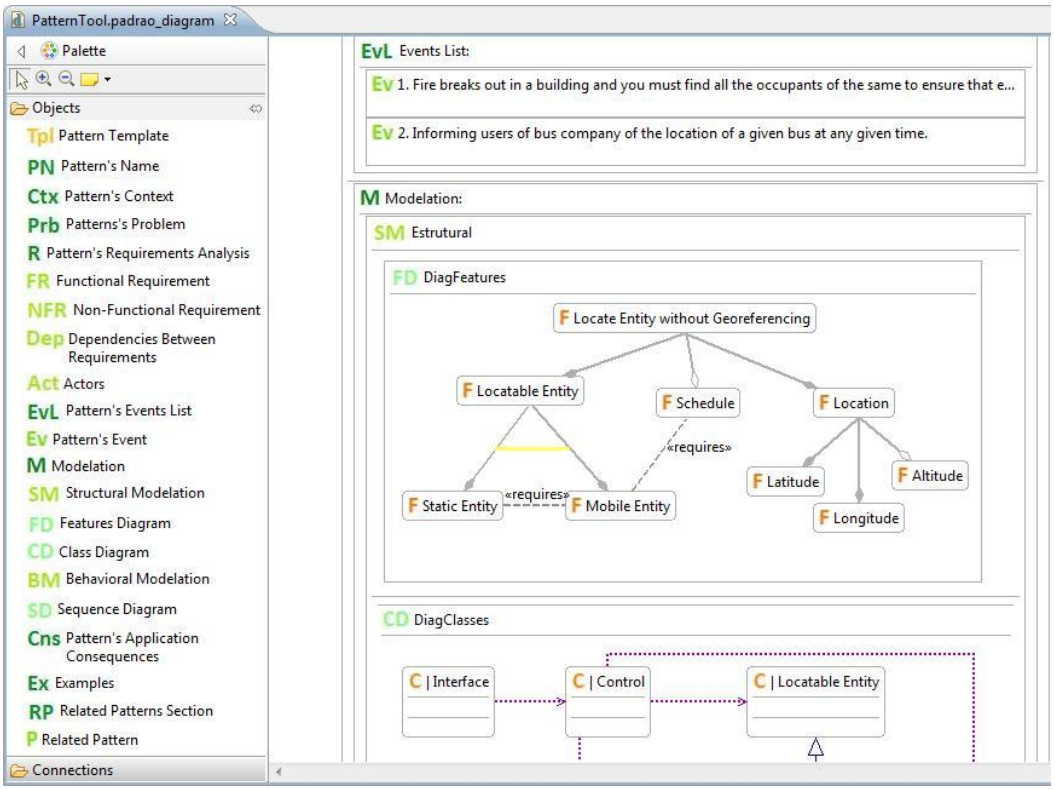

Figure 9. Generic View of the Pattern Tool

The pallet of the tool is presented on the left hand side, where all the fields needed to create a pattern can be added. The fields are placed on the pallet in the order wherein they must be placed in the template. The colors help to understand when a field should be placed within another. The Pattern Template field should be the first one to be clicked, as it starts the creation of a new pattern. The hierarchy of the fields of the template is represented by a color scheme. For example, the fields to be filled within this Pattern Template are icons with dark green color, such as the Pattern's Name field, Pattern's Requirements Analysis, among others. When a field is placed within a field with a Dark Green icon, its icon is Light Green, and finally, when a field must be placed inside a field with a Light Green icon, its icon color will be an even lighter green, as is the case of the Features Diagram icon.

To edit the feature, class and sequence diagrams, after inserting its compartment in the template, an additional editor can be opened, which can be used to build the diagram. The diagram's editing pallet is organized like the template's pallet. We can see the example of the Features Diagram's Editor in Fig. 10.

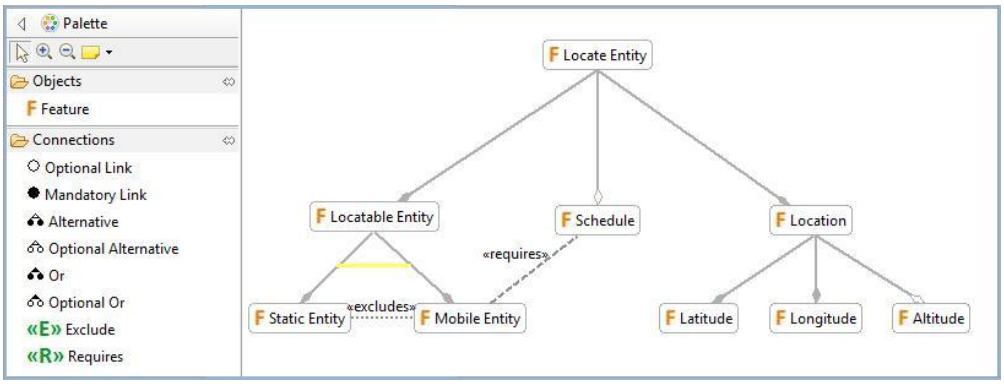

Figure 10. Features Diagram's Editor

5.1 Pattern Tool Evaluation

For the evaluation of the Pattern Tool we selected 15 subjects, all students of the MSc on Informatics Engineering of our Faculty. Only 2 had already finished the course and were working in industry. All of them had attended courses on analysis models. About 85% had attended a course on GIS.

The tool evaluation consisted of a set of questions about language expressivity and syntax, tool usability and the satisfaction level of the users.

The evaluation results were positive: the users considered the tool very useful and intuitive, easy to use and understand. The subjects were asked the questions presented below. The aim of the questions is described and the results of the evaluation are also discussed:

1. How easily were the concepts identified in the tool? The aim was to assess the quality of the concepts representation. Concerning the representation, 10 out of 15 users thought it was "very easy" to identify the concepts, while the remaining 5 considered it "easy".

2. What is your overall impression of the tool? The aim was to evaluate whether users liked to use the tool. In general, results were positive as 12 out of 15 users thought the tool was "good", two rated it "very good" and one of them considered it "average".

3. Do you consider that it was easy to migrate the pattern from paper to the tool? The aim here was to evaluate whether users felt lost while moving the pattern from paper to the Pattern Tool. The results showed that 11 out of 15 users considered the migration "easy" and 4 others considered it "very easy".

4. Do you consider the tool useful? The aim was to evaluate whether users felt that the tool was useful. The results were positive. They said that no other tool enabled the creation of the pattern with such detail.

5.1.1 Evaluation Threats

Having an evaluation in the industrial/business environment would give a different perspective to the evaluation of the tool. However, the users who evaluated both the tool and the description of the pattern knew the most recent technologies, which may not be frequent in a business environment. The evaluation was performed with only 15 users but, although the statistical significance is reduced, the results are indicative of the acceptance of the evaluated approach.

6. RELATED WORK

Oliveira *et al* (2010) presented an aspect-oriented approach to model Web GIS applications. The author developed a model to identify, modularize and compose crosscutting concerns, more precisely spatial concerns, in an application, to tackle low level modeling in Web GIS applications, which can lead to scattered components, creating difficulties at the modularity level and compromising the reusability of the system. The application of this model offers advantages such as modularity, efficiency, extensibility and reuse of applications. The identification of requirements is achieved through a thorough knowledge of the Web GIS domain, obtained by domain analysis techniques. Although this work used AOSD techniques for modeling Web GIS, it did not consider reusing patterns.

Gordillo *et al.* (1999) developed a technique for modeling object-oriented GIS where, from the basic geographic model, spatial characteristics are added to each object in a dynamic and transparent way. The work demonstrates how reuse can be useful in this kind of application. This technique relied on object-oriented methodologies in order to obtain reusable, modular and modifiable software, as well as objects that encapsulate knowledge, which can be adapted to different needs. Also, design patterns were used as a powerful strategy for developing good design solutions to recurring problems. This work focused on design and did not use aspect techniques.

In (IJDE, 2008), a catalogue of common functionalities for defining a basic Web GIS application is proposed. However, the description of the functionalities is not as detailed as the approach presented in this paper.

7. CONCLUSION

This paper presented an approach based on aspectual analysis patterns, which was applied to enable the reuse of spatial concerns specifications in Web Geographic Information Systems, with the aim of maintaining and updating both the structural and the behavioral models of these applications, and trying to avoid flexibility and reuse problems. Web GIS applications were examined to identify some of their volatile but reusable spatial concerns, since these applications are characterized by the constant change of requirements.

Thus, an aspectual analysis pattern template was defined (and adapted from previous work) and applied to the modeling of spatial concerns patterns. One of the advantages of the template is the use of the MATA notation, an aspect-oriented modeling technique, which provides efficient mechanisms for modeling and composing static and behavioral elements of a pattern.

The spatial concern example “Show a Geo-referenced Entity” was used and its analysis pattern was described. Also, a tool for patterns description was created to complement this work and applied to a real case study (the CLIP system).

Some work is still under development, which includes the analysis of additional patterns, identified for other activities, such as the Presentation of Geographical Data or the Insertion of Temporal Constraints in Geo-referenced Objects. The tool also needs improvement, like the development of a catalogue of spatial patterns, that is, a list of patterns which can be updated and re-used as needed in the development of new applications.

ACKNOWLEDGEMENT

Thanks to CITI and project ASPECT_WEB for partially funding this work.

REFERENCES

Dragicevic, S., 2004.The Potential of Web-based GIS.*Journal of Geographic Systems*, Springer-Verlag, Vol. 6, pp. 79-81.

Filman, E,Elrad, T., Clarke, S. and Aksit, M., 2005. *Aspect-Oriented Software Development*. Addison-Wesley.

Fowler, M., 1997.*Analysis Patterns - Reusable Object Models*. Addison Wesley.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J., 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*.Addison-Wesley.

Gordillo, S., Balaguer, F., Mostaccio, C. and Neves, F., 1999.Developing GIS Applications with Objects: A design Patterns Approach.*GeoInformática*, Vol. 3, No.1, pp. 7-32.

Hamza, H. and Fayad,M. , 2004, Applying Analysis Patterns Through Analogy: Problems and Solutions, *Journal of Object Technology*, Vol. 3, No. 4, pp. 197-208.

IJDE, 2008.Digital Earth Summit on Geoinformatics: Tools for Global Change Research. *International Journal of Digital Earth*, Vol. 1, No. 1, pp. 171 -173.

Kang, K., Cohen, S., Hess, J., Nowak, W. and Peterson, S., 1990, Feature-Oriented Domain Analysis (FODA) Feasibility Study. *Technical Report: CMU/SEI-90-TR-021*, Pittsburgh, USA.

Oliveira, A., Urbieta, M., Araújo, J., Rodrigues, A., Moreira, A., Gordillo, S. and Rossi, G., 2010. Improving the Quality of Web-GIS Modularity Using Aspects, in *Proceedings of QUATIC 2010*, pp. 132-141.

Pantoquilho, M., Raminhos, R. and Araújo, J., 2003. Analysis Patterns Specifications: Filling the Gaps, *Viking PLoP 2003*.

Silva, S., Araújo, J., Rodrigues, A., Urbieta, M., Moreira, A., Gordillo, S., Rossi, G., 2012, Reuse of spatial concerns based on aspectual requirements analysis patterns, RCIS 2012, pp. 1-6.

Sommerville, I., 2010.*Software Engineering*, 9th edition.Addison Wesley.

Whittle,J., Jayaraman, P., Elkhodary, A., Moreira, A. andAraújo, J., 2009, MATA: A Unified Approach for Composing UML Aspect Models Based on Graph Transformation. *Aspect-Oriented Software Development VI 6*, pp. 191-237.