Innovations Syst Softw Eng
DOI 10.1007/s11334-013-0206-y

SI: QUATIC 2010

Web-GIS models: accomplishing modularity with aspects

Matias Urbieta · Ana Oliveira · João Araújo · Armanda Rodrigues · Ana Moreira · Sílvia Gordillo · Gustavo Rossi

Received: 3 July 2011 / Accepted: 12 April 2013

Abstract Spatial concerns of Web geographical information systems (Web-GIS) are inherently crosscutting and volatile: crosscutting because they affect multiple functionalities of Web-GIS systems, and volatile because their status may change often. If these concerns are not modularized properly, the quality of Web-GIS services, particularly with regard to adaptation and evolution, can be severely compromised. This paper uses aspect-orientation to model crosscutting and volatile spatial concerns. By modeling both types of concerns, crosscutting and volatile, as candidate aspects, one can use dynamic weaving to add or remove them from a system at runtime. The aspect-oriented approach proposed starts with the identification and specification of crosscutting concerns and follows by composing these using modeling aspects using a transformation approach, an aspect-oriented modeling technique. The conflicts that can emerge due to the composition order are also taken into consideration. Finally, this paper proposes a set of reusable GIS crosscutting concerns, documenting them in a concern catalogue.

Keywords Web geographical information systems · Aspect-oriented software development

M. Urbieta (✉) · S. Gordillo · G. Rossi
Lifia, Facultad de Informática, UNLP, Buenos Aires, Argentina
e-mail: matias.urbieta@lifia.info.unlp.edu.ar

S. Gordillo
e-mail: gordillo@lifia.info.unlp.edu.ar

G. Rossi
e-mail: gustavo@lifia.info.unlp.edu.ar

M. Urbieta · G. Rossi
CONICET, Buenos Aires, Argentina

A. Oliveira · J. Araújo · A. Rodrigues · A. Moreira
CITI/FCT Universidade Nova de Lisboa,
Monte de Caparica, Portugal
e-mail: aidm.oliveira@gmail.com

J. Araújo
e-mail: joao.araujo@fct.unl.pt

A. Rodrigues
e-mail: a.rodrigues@fct.unl.pt

A. Moreira
e-mail: amm@fct.unl.pt

A. Oliveira · J. Araújo · A. Rodrigues · A. Moreira
Departamento de Informática, CITI, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Lisbon, Portugal

1 Introduction

In the domain of Web geographical information systems (Web-GIS), spatial information is constantly added and updated (even by final users). Additionally, new requirements, involving spatial functionality, emerge constantly and some of them are volatile, that is, they are only temporarily required. For example, some features are tested with users and eventually eliminated if they prove not to be frequently used. As a consequence, design tends to be more complex. An interesting example of this case is shown in Fig. 1 where we can see "Additional Information" to a photograph of Lisbon, on a Flickr home page.[1] This information, highlighted with a box in the page, shows the place where the picture was taken. (Geo-referencing of photos was originally added by users using GreaseMonkey[2] technology and later Flickr started to provide it as well.)

Adding new requirements into an application often becomes a hard task to the development team, especially

[1] http://www.flickr.com/.

[2] https://addons.mozilla.org/en-US/firefox/addon/748.

Published online: 15 May 2013

when these new requirements introduce scattered and tangled code into the application, compromising the quality of the system's modularity. This results in an increased difficulty in software maintenance, evolution, and adaptability. A reasonable way to deal with this issue is to encapsulate scattered and tangled behaviors in separate modules using aspect-oriented techniques [15]. Hence, we use aspect-orientation to modularize tangled and scattered behaviors—that is, *crosscutting concerns* —in Web-GIS. Crosscutting concerns are later composed, or weaved, in different points, or joinpoints, of the same or other applications.

This paper focuses on spatial concerns, which are inherently crosscutting and volatile. To deal with crosscutting concerns, we will, therefore, use aspect-orientation, providing a better modularization mechanism to specify spatial concerns.

Typical crosscutting concerns arise when dealing with spatial data in Web software (e.g., the user's location). The market increasing interest in mobile technology, associated with the need to improve applications that are able to consider the user's real position, becomes stronger every day. Web-GIS applications tend to be complex as they combine the volatile nature of Web software with the inherent complexity of dealing with spatial data. Moreover, from a software engineering perspective, location-aware behavior, typical of Web-GIS, usually cuts across other application concerns, since this type of behavior is likely to have an impact across different application features [9,28].

We aim to analyze how geographic requirements may interfere in the design of Web applications [33]. More specifically, our goal is twofold: (i) to develop a characterization of these application functionalities that may be affected by geographic crosscutting requirements and (ii) propose an aspect-oriented approach to handle them, improving modularization.

The approach proposed here is based on aspect-orientation and uses modeling aspects using a transformation approach (MATA) [40] to specify aspects. We will demonstrate how spatial behavior can be isolated from other concerns to improve modularity in our application domain and, after having them modularized, how volatility is controlled in the composition phase by plugging and unplugging concerns. This work builds on initial ideas presented in a previous paper [31]. Here we explore those ideas further and propose solutions to contemplate aspect interaction of spatial concerns.

The remaining of the paper is organized as follows. Section 2 presents a motivating example. Section 3 describes some background of our work. Section 4 introduces potentially reusable spatial concerns in Web applications, creating a catalogue described in a pattern-like style. Section 5 presents our approach and Sect. 6 applies it to a running example, discussing the results obtained. Section 7 discusses how an abstract and system-independent knowledge base can be built for Web-GIS. Section 8 discusses advantages gained with the separation of GIS concerns from modeling to implementation and Sect. 9 presents related work. Finally, Sect. 10 draws some conclusions and describes future work.

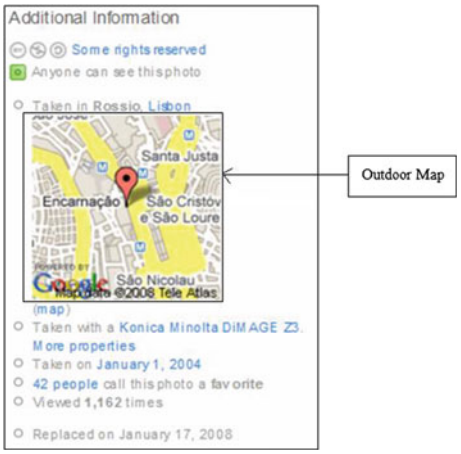

Fig. 1 Flickr page for a photograph with Outdoor map information

2 Motivating example

A typical new requirement that may be added to a Web-GIS application is the "indoor representation". Figure 2 shows a Web application for a shopping centre[3] that provides Indoor Map support for presenting store locations. The application offers a "search function" to provide information about stores such as their address and their location inside the shopping centre (indoor position)—using the map pointed with an ellipse. An Outdoor Map is also shown, indicating the location of the shopping centre in a global map. Both examples in Figs. 1 and 2 reveal how the same GIS requirements may be present in applications of different nature, contemplating many occurrences during one execution of the same application.

The inclusion of the new requirement "indoor representation" creates scattered and tangled concerns in the application among its core, or base, concerns. To make matters more difficult, adding other concerns increases the application complexity compromising its maintenance. (Section 5 describes how to deal with these new requirements in more detail and Sect. 6 illustrates some examples.)

Figure 3 illustrates how the introduction of new requirements might impact the application's structure. This figure shows a UML sequence diagram for "Show Store". This is taken from the Web-GIS application Maps@Web [12]. It contains tangled behavior as a consequence of composing several GIS concerns such as indoor representation, location sensing and points of interest. We can see that the behavior of each requirement is scattered along the sequence diagram and

[3] http://www.colombo.pt/.

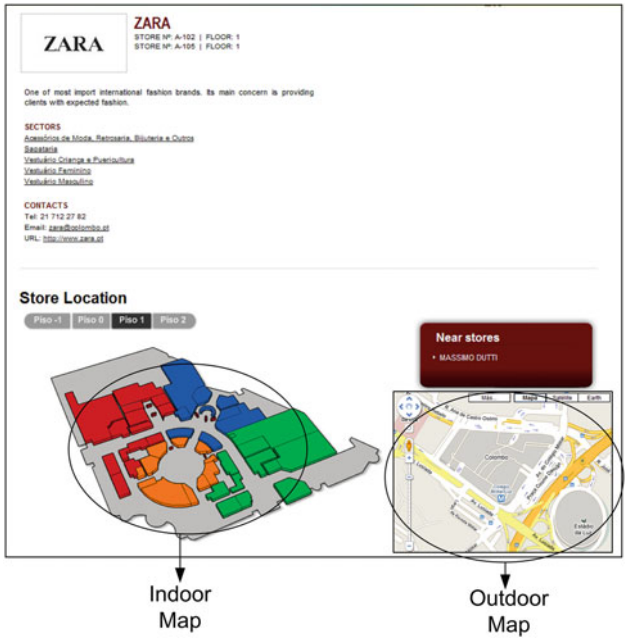

Fig. 2 A shopping portal with Indoor and Outdoor map support

tangled with other behavior, characterizing crosscutting situations. For example, location-aware crosscuts indoor and outdoor representations, due to the current user position, which can be either in a global map or in a specific building.

3 Background

Modeling of Web-GIS generally involves the identification of requirements and system functionalities. It is of major importance to consider the modularization of crosscutting concerns in these applications, as it can be seen in [26,42].

In this context, *aspects* were introduced to modularize *crosscutting concerns*, which could not be modularized using object-orientation [25]. Thus, crosscutting concerns are spread, or cut across, other concerns, creating tangled and scattered representations of the program that are difficult to understand and maintain [35]. Aspect-oriented software development (AOSD) [15] appeared to handle crosscutting concerns in all stages of the software lifecycle. Aspects appeared first at the programming level, with AspectJ programming language [4], an aspect-oriented extension of Java. The main concepts included in AOSD are *aspects*, *joinpoints*, *pointcuts* and *advices*. A *joinpoint* specifies a well-defined point in the execution of the base program that will be affected by an aspect, such as a method call, an access to a variable, etc. A *pointcut* specifies a set of joinpoints and data associated to them. An *advice* defines code that can be executed when a joinpoint is reached in the program execution.

Currently aspects are used across the whole lifecycle. In particular, several aspect-oriented requirements engineering

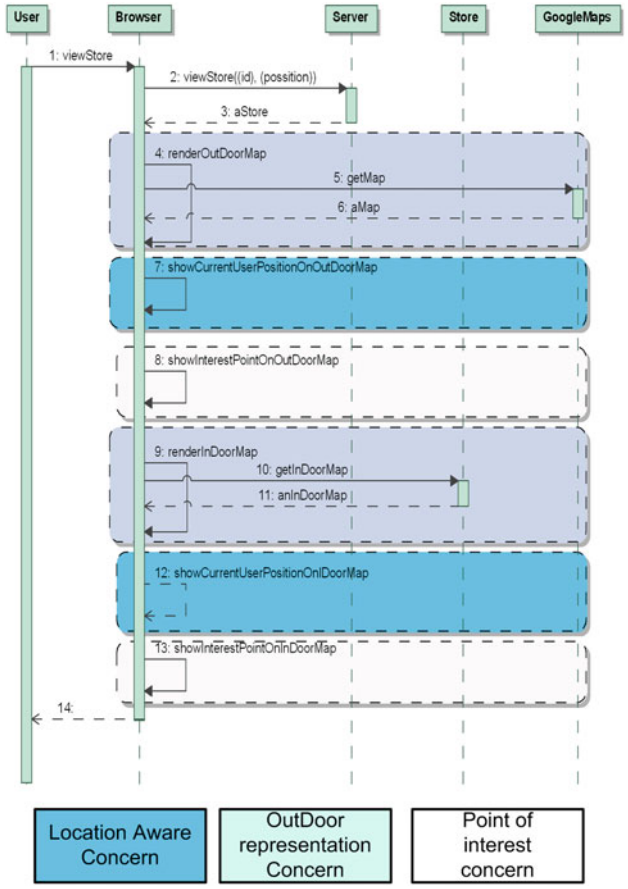

Fig. 3 Sequence diagram with tangled behavior due to crosscutting concerns

approaches have been proposed, such as [7,27,35], AORE with use cases [21,22], Theme [5] and MATA [40]. Only MATA is described here, given both its high expressiveness [41] to model and compose crosscutting behavior, and because it uses graph rules which allow more composition possibilities and the identification of some aspect interactions.

3.1 MATA

The MATA aspect-oriented modeling tool is based on UML, allowing aspects composition using class diagrams, sequence diagrams and state diagrams. Here we focus on MATA to model aspectual scenarios by using and adapting sequence diagrams. To specify aspectual scenarios, three new stereotypes where created to define composition rules:

- ≪create≫ which states that the element will be created in the base scenario
- ≪delete≫ which states that the element will be deleted of the base scenario
- ≪context≫ which states that the element will not be affected by the other two stereotypes.

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 scenarios, 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 scenario, always respecting the composition rules defined in the aspectual scenario. The resulting composed scenario describes the behavior of both scenarios, according to the rules defined. MATA allows more composition combinations than other existing aspect-oriented modeling tools [11] and also the identification of some aspect interactions.

Figure 4 shows an example of a MATA rule defined in the sequence diagrams context. R1 specifies that the aspectual behavior consists of an interaction between two objects that must be instantiated to two objects in the base. The rule says that the fragment par (that specifies parallelism) and messages r and s in one of the sections of the fragment are created, i.e., they define the aspectual behavior that must be inserted in the base. However, since p is defined as "≪context≫", it must be matched against a message with the same name in the base. The resulting composed model when applying R1 is shown on the top right-hand corner of the figure. Note that since *q* and *b* are not part of the rule they come after the par fragment.

The starting point of the approach introduced in this paper is the identification and specification of crosscutting concerns, which is followed by their composition using the MATA language and sequence diagrams.

4 Spatial concerns in Web applications

We consider a spatial concern as a special kind of software concern that refers to the set of requirements that deal with geographical features of an application, such as locations, maps and routing algorithms.

Geographical information is characterized by having spatial attributes, like a geographical position, which may be global (e.g., locations captured by a GPS) or local (e.g., inside a building). For this analysis, we assume the availability of typical and basic geographical resources [3,26] such as maps, graphs, layers, etc., provided and managed by an open source geographical server such as OpenStreetMap [32]. Instead of focusing on individual geographic requirements, we group them in sets of related requirements dealing with the same matter of interest that as mentioned before are called spatial concerns.

4.1 Typical spatial concerns

This section characterizes the most common types of spatial concerns. For each type we provide guidelines to facilitate their identification and simplify their realization in Web software. As shown later in Sect. 5, we propose that these concerns be used as patterns, for achieving modularized design solutions through the reuse of their specification and implementation recommendations.

We classify spatial concerns by focusing on the kind of impact they have on the underlying application. To improve understanding we will focus on the type of applications described in Sect. 1, though most of the discussion can be applied to a broader range of applications. For each type of spatial concern we present a template composed of five fields: name, description and example, impact and recommended solution. The meaning of each field will become evident in the text.

4.1.1 Spatial business object

Description and example

This kind of concern is commonly faced when enhancing applications adding some sense of location to business objects, as well as the corresponding spatial functionality to manipulate this location. For instance, a bus service management system can be improved by providing real-time bus locations, offering more precise and timely information to managers and passengers. This requires adding location and estimated arrival time to the original bus stop map.

Impact

This problem involves the introduction of spatial operations (to compute distances) and location information to

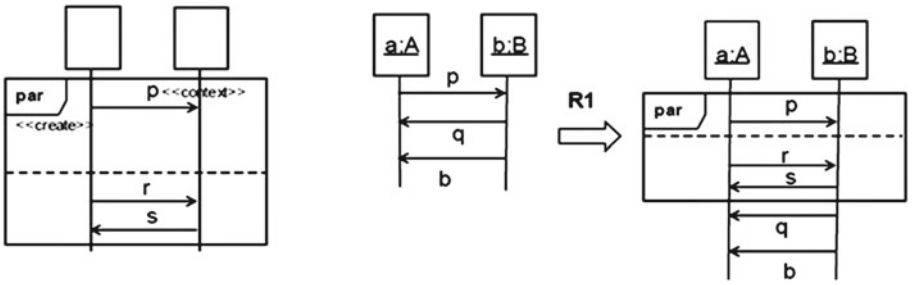

Fig. 4 An example for a MATA rule

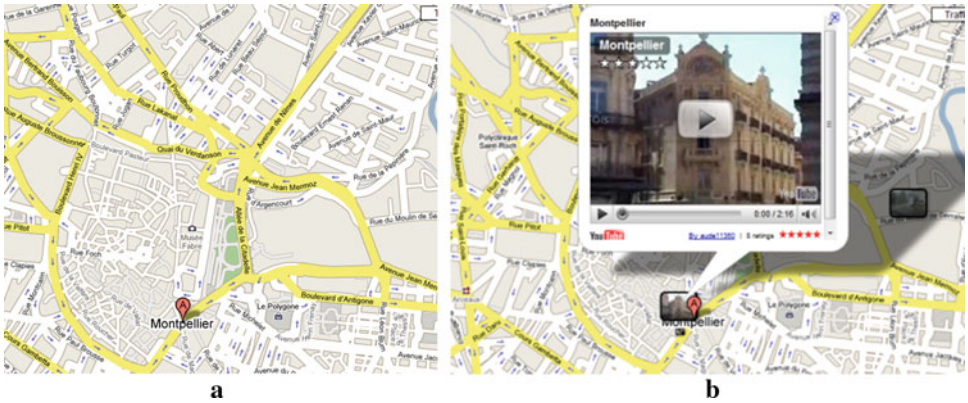

Fig. 5 **a** Simple Web application with road information. **b** After adding support to associate and play videos

a b

describe the object's spatial features. The latter may consist in enriching the object with latitude and longitude variables or with a variable pointing to a full-fledged location entity. In the example of the Bus object that becomes location-aware, we might have tangling code because new presentation logic is demanded for adding a map to the bus information view (and the same happens in every similar example). Additionally, to the operation that returns the current bus position, new business logic may be appended to compute the arrival time to a given bus stop. Moreover, if we have several location-aware objects, such as ferries or taxies, the code for supporting this functionality becomes scattered. The introduction of location-aware requirements also demands new user interface widgets for presenting the location information in a map.

Recommended solution

Location information enriches an application object either by making available latitude and longitude variables or by adding a reference to a more complex entity representing location data. There are two possible solutions:

- Decorating the domain object: using the decorator pattern [16], the domain object can be extended with either the pair of variables or the reference to a "Location". This will demand creating a new object class that wraps the original target domain object.
- Introducing variables into the domain object: using aspect-oriented techniques [15], the domain object is extended by means of an aspect that introduces the variables or the Location entity in the class. The aspect registers the variables or the references extending the class, by means of a weaving process.

Both alternatives provide a non-intrusive solution because none of them demands any change on the original domain object.

4.1.2 Rich spatial data

Description and example

This concern category describes a usual situation that requires enriching a geographical object with new object information regarding to non-spatial characteristics and provide ways to manipulate such information. A typical example of this type of concern arises when adding videos to a specific location on a map. Figure 5a shows a simple Web application (in this case based on Google Maps) in which a new set of requirements to support the association of videos to spatial positions, their display, edition, etc., is added (Fig. 5b).

Impact

Already defined data structures must be modified or new ones may need to be created to support this new kind of functionality. Additionally, the application's linking structure may be modified. Finally presentation issues must be adjusted (e.g., by introducing rich interaction support). We might also experience tangling when including the logic that enables the handling (adding or removing) of movies for specific locations.

Recommended solution

Enriching a geographic object implies augmenting the object with either a variable referencing the new descriptive information, or with methods (setter and getters for variables) implemented simply using a Decorator object or aspect introduction techniques. The former defines an object that wraps the original one, with the new characteristics and delegating method invocation to the target object. The latter introduces variables and methods for referencing the new descriptive information by using the "introduction" feature provided by aspect engines.

4.1.3 Spatially constrained behavior

Description and example

This situation arises when we need to change or modify the behavior of an object according to its actual geographic location. For example, in a real estate agency, different houses are shown in a city map (the houses are business objects enriched with spatial information). As customers get interested in the taxes to be paid when owning the house, the agency must add a new behavior that computes the tax according to the geographical area where the house is located. The situation

might get more complicated if some tax is only applied during a certain period of time (e.g., as a consequence of an unusual meteorological event).

Impact

The introduction of new (geographically constrained) business logic may either involve a complete set of methods or enhance already defined ones. These changes may generate tangled code, when combined with existing algorithms, such as other taxes, and may generate scattered code, when the same logic is introduced within different object types. For example, when a zone suffers an inundation, the government may promote a tax reduction policy for those fields and buildings affected by the catastrophe. The behavior for computing the tax may be scattered in those objects responsible for tax computation. In an object-oriented model, the Field and Building objects could be responsible for their own tax computation. Alternatively we could delegate the computation to strategy objects but we would still have a conditional statement on location variables to configure the strategy.

Recommended solution

Any interception technique such as Wrapper objects or aspect-based interception (using pointcuts) allows dealing with this kind of improvement. Using aspects offers more flexibility, since they can provide pre-invocation, post-invocation and in-invocation enrichments, using the corresponding before, after, and around pointcuts. A Wrapper can be used for intercepting a method call and enrich or override the expected functionality.

4.1.4 Map adjustments

Description and example

This kind of spatial concerns is used for extending or restricting the available spatial business objects according to the application's constraints. The spatial data available in a Web application may be restricted or extended according to a specific concern, which may imply that certain parts of a map are unavailable or are useless for specific operations or services. These types of extensions or restrictions may be temporal or permanent. When geographic objects un/availability is temporal, status calculation can be done in real-time and may require collaboration with other artifacts, components, or external systems. This type of concern may arise in (at least) two different ways: augmentation and restriction of the available geographical data. Next we present different examples of each kind.

Augmentation In Portugal, the Tejo River separates the country's Capital, Lisbon, from the Almada town, where the FCT campus of Universidade Nova de Lisboa is located. To go from Lisbon to the FCT campus using public transport, we may use the ferry line. However, if the ferry trajectory is not included in the map we will not be able to find a path including it and thus a longer path will be provided. More volatile requirements may arise when roads are available only for a limited period of time (during sports events, for example), such as the case of some of the Paris Dakar rally roads—probably not available before—the Lisbon marathon—overriding roads direction in the marathon time, etc.

Restriction Disabling parts of a map temporarily for path searching is quite common when there are streets under repair, public demonstrations, or events that make roads unavailable. These kinds of restrictions can also be applied to indoor representation, e.g., having a lift unavailable, blocked areas of a building, etc.

Impact

A concrete and very direct impact arises in algorithms that manipulate spatial data; specifically, the respective requirements lead to changes in path-finding features. For example, in the case of adding new transport services, computing node adjacencies (segments in a graph) should change, since new edges, such as public transportation and highways, must be taken into consideration. This introduces scattered and tangling code at each place where a node is asked for its adjacencies, because each search algorithm defined in the application is affected by the requirement. In the case of blocked or unavailable streets, the graph is modified, by removing the corresponding edges. Applying graph changes directly to the corresponding database may be unmanageable when the nature of changes is volatile. For example, the responsibility for determining whether a given street segment must be pruned or not can be delegated to a third-party service provided by the highway administration. In this case, it is unreasonable to modify the database because the information is volatile and only available on demand.

In both cases, the corresponding presentation layer must be modified to show new or blocked streets information. This is important because the user must be aware that the route is different from the normal route and must realize that there is no bug in the results.

Recommended solution

Usually, nodes of a layer are resolved by means of a data access object (DAO) [1] or service when invoking a given method. Therefore, the augmentation or reduction of this working set of nodes must be executed when the solution method is called. Following the ideas presented in the solution for "Map Adjustments", by means of Wrappers or Intercepting aspects, we can process a posteriori a given method call to expand or reduce the returned set of available nodes. That is, when adding a new layer such as ferry line, the DAO method that resolves nodes can be intercepted in order to append the ferry line nodes to the corresponding method invocation result.

4.1.5 Temporal spatial objects

Description and example

A specific case of "Map Adjustments" that is worth to be treated separately occurs when geographical objects can be temporally constrained. For example, the road network may change several times during the day, e.g., some streets become available or unavailable according to the time or the day of the week.

Impact

This kind of requirement may generate tangled code in all spatial object types, which are subject to temporal restrictions, for adding the needed structures that describe both the temporal availability and temporal information management functions.

It also generates tangling code at any place where the object's collaboration is demanded, e.g., any spatial data resolver object which is responsible for loading objects from the proper repository, will be tangled with the necessary temporal checks to resolve/identify the objects which should be available in the specific contextual period of time.

Recommended solution

Since this is a special case of "Map Adjustments", the same solution can be applied. By intercepting method calls or decorating methods, the logic that restricts the object behavior based on a timetable can be appended to the base object behavior. The timetable may define periods of time where the object is available.

4.1.6 Personalized geographical objects

Description and example

Concerns of this sort help changing the availability of geographical objects according to the user's profile. Web applications may change its behavior depending on the user's profile, for improving the user experience. This may imply changes in the data sets' availability conditions, as described in "Map Adjustments".

For example, depending on the current user's means of transport, a path finding algorithm can calculate the best route in different ways. For example, using a bike as transportation excludes highways or bus lanes. A similar example occurs when we plan a route for disabled people where stairs must be skipped for indoor and outdoor routes.

Impact

Again, this kind of requirement generates tangled code in path-finding algorithms, since some street segments may be inadequate for the user's current means of transport. For instance, a truck may not be allowed downtown during working hours. Additionally, tangling may also occur in other application components, such as the presentation components (e.g., to explicitly symbolize pedestrian or cycle lanes which cannot be used by trucks).

Recommended solution

Wrapping the algorithms with decorators as in "spatially constrained behavior" or with an aspect-oriented technique by weaving the adaptation code transparently can trigger profile-based adaptation. Additionally, the interface has to be modified to incorporate the new variants.

4.1.7 Geographic interfaces

Description and example

When a spatial concern arises, it usually introduces changes in conceptual models requiring modify or upgrade the user interface of geographic objects in order to reflect such changes. Though not strictly a spatial concern, it is clear that most of the previously cited examples might introduce changes in the application's user interface, specifically in the geographic objects (e.g., maps). Adding temporal availability to spatial objects ("Temporal Spatial Objects") or adjusting the spatial dataset availability ("Map Adjustments") are changes related to spatial objects' behavior. However, as previously mentioned, a lack of a suitable presentation may produce a misleading perception of the application functionality. For example, if blocked streets are taken into account in a path search, they must be appropriately visualized in the map (for instance using a standard red color).

Impact

The presentation layer needs to be improved in order to provide a proper presentation for spatial concerns such as: map widgets when enabling spatial behavior to business objects ("Spatial Business Object"); labels and pointers over map widgets when adding temporal availability to spatial objects ("Temporal Spatial Objects") that notify to users reasons of objects unavailability and labeling availability and criteria constraints on the relevant spatial dataset ("Map Adjustments").

Recommended solution

Since Web application interfaces can be developed using different approaches such as declarative (HTML) or programmatically defined [13,18], the solution depends on this choice. In the former case, using XSLT transformations, we can modify the HTML document model for including markups or Javascript code. In the latter, using a Decorator object will help to wrap and enhance a programmatically defined interface. Later on, we present an example that uses interface transformations for solving this change.

4.2 Discussion

The previous classification covers an exhaustive set of common types of concerns related with geographic information in the context of Web software. The conclusions we can draw are that:

- most concerns, if not all, introduce code tangling or scattering in other concerns. We can go further in this analysis to see that some specific concerns are more affected than others (e.g., path finding and manipulation). Consequently, if we want to avoid the well-known problems that arise when dealing with tangled or scattered code [15], a good alternative is to use a modularization mechanism to allow us to keep these concerns separated, and treat them independently during the development process;
- as it was shown in the examples, Map Adjustments also appears to be a volatile concern, thus complicating future maintenance, but can also treated using AOSD mechanisms.

4.3 Identifying spatial concerns

This sub-section outlines an informal strategy to identify spatial concerns in Web applications. Their identification and isolation will improve the level of modularity of the application reducing other concerns' complexity; therefore making the whole system easier to maintain. We propose an incremental process for identifying spatial concerns by evaluating the following questions:

(a) Does a business object need to be geographically located (Sect. 4.1.1: Spatial Business Object)? This can be accomplished by associating the attributes "latitude", "longitude" and "altitude" to the business objects, or else be achieved by some automatic process of geo-coding. Additionally, the objects considered in the application should include spatial behavior. This involves the use of the objects' locations to tailor behavior that will enrich the logic of both the objects and the application as a whole.

(b) Is any business object being enriched with additional information (Sect. 4.1.2: Rich spatial data)? This involves connecting several types of information to geo-referenced business objects, considered relevant in the context of the application. This type of concern can only be identified if step (a) has already been completed.

(c) Does any business object behavior suffer customizations based on its spatial contexts (Sect. 4.1.3: Spatially Constrained Behavior)? This involves identifying and associating specific behavior to objects, depending on where they are located. This concern can only be identified if (a) has already been completed.

(d) Are spatial objects being enabled and disabled affecting its availability (Sect. 4.1.4)? This involves altering the relevance of geo-referenced business objects pertaining to the spatial dataset currently in use, and executing these alterations during the execution of spatially enabled algorithms. This concern can only be identified if (a) has been completed already. Moreover, it may be refined if the changes in the available spatial data are the result of restrictions, temporal or others (Sect. 4.1.5).

(e) Are spatial object taking into account the user/s role for customizing its behavior (Sect. 4.1.6)? This involves the need to categorize users in the application's context. The identification of different types of users may generate differences in terms of the relevance of spatial data and behavior, towards these users. Any of the spatial concerns identified previously may be affected by profiling.

(f) Are spatial objects or related information being presented to users (in GUI, reports, etc.) (Sect. 4.1.7)? This involves the spatial presentation of information and behavior contemplated by the application and the objects involved in it. In this context, this concern interacts with all the spatial concerns previously defined (from (a) to (e)).

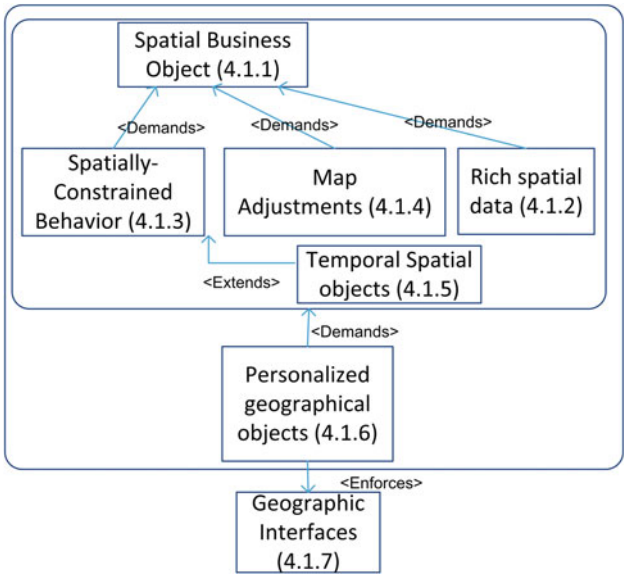

Fig. 6 Relationships among spatial concerns

Figure 6 shows how the different characterizations described in the previous questions are related using binary relationships. To qualify these relationships, a set of stereotypes was defined. The possible stereotypes are ≪*Demands*≫, ≪*Extends*≫ and ≪*Enforces*≫. First, ≪*Demands*≫ describes a dependency restriction that requires the presence of the target characterization when the source is detected. For example, when the enrichment of an object behavior is intended through the specification of its location, the location features required by the enrichment logic must be provided. Secondly, ≪*Extends*≫ indicates a specialization for a given characterization. That is, adding temporal availability to spatial objects corresponds to a special case of adjusting the spatial dataset's availability because it restricts the available data according to a timetable. Finally, ≪*Enforces*≫ indicates that a given characterization triggers the target characterization.

In the following section, we present an aspect-oriented approach that takes into account the characterizations presented, covering from requirements modeling to implementation. The approach helps to design solutions for the kind of requirements presented so far. Our approach aims at providing a conceptual tool for an early detection of crosscutting spatial concerns, detecting tangling and scattering, and providing better solutions by isolating these concerns as separate components. Thus, we seek to improve the level of modularization offered by traditional object-oriented and component-oriented approaches. To facilitate comprehension, we start with some background on aspect-orientation.

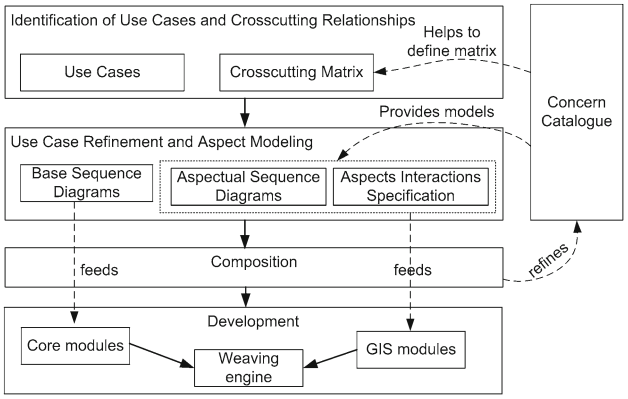

Fig. 7 Modeling GIS concerns with aspects

5 Modeling spatial concerns with aspects

So far, we have identified and characterized the most common GIS concerns describing their typical features. Next, we present an approach to analyze, design and implement GIS concerns.

Figure 7 defines a process for modeling spatial concerns using aspects in Web-GIS applications. The process consists of four general activities: identification, specification, composition and implementation. In the context of this paper, we will refine each of these activities to scenario modeling, where use cases and sequence diagrams are the techniques used.

The first activity, identification of use cases and crosscutting relationships, identifies the use cases and represents the relations among crosscutting use cases using the stereotypes ≪include≫ and ≪extend≫. That is, a use case that is included in several use cases or a use case that extends several ones is considered crosscutting. In addition, we adopt the stereotype ≪invokes≫ [36] to be used when a use case activates one or more use cases. Although some works do not recommend describing dependencies between use cases, such as [24], we believe that pointing out use case relationships helps detecting crosscutting concerns, as it will be described next in the Use Case Refinement. The strategy discussed in Sect. 4.3 may be used to help identifying use case relationships.

Our use case diagram is complemented with a crosscutting matrix that is used to detect possible crosscutting behavior between the use cases. Also, a "concern catalogue" can be used to contribute:

- to identify already recognized crosscutting concerns, but also contributions, or dependencies, between them and,
- to give feedback to logged concerns, enriching the concern information as well as its known-issues learned from experience.

The second activity, use case refinement and aspect modeling, refines the use cases adding more detailed information and designs the crosscutting behavior between concerns taking into account the crosscutting matrix. The first task is to design the non-aspectual use cases (called base use cases) using UML sequence diagrams (that describe base scenarios). The second step comprises the modeling of aspectual use cases using UML sequence diagrams enhanced with MATA graph rules (i.e., aspectual scenarios). When a concern stored in the catalogue is recognized and selected for instantiation in the identification step, its consolidated models are reused profiting from full-fledged concerns.

From the relationships detected in the crosscutting matrix, we must identify the interaction and possible conflicts among aspects giving as result a consistent set of joinpoints. The aspects coexistence can lead to incorrect results when the aspects are composed. That is, an aspect may conflict with another; some of such information can also be taken from reusable catalogues. This occurs when an aspect changes models in such a way that it prevents the application of another aspect. Jayaraman et al. [23] propose an approach that can be used for detecting this situation where, by means of a critical pair analysis [20] of MATA models, some aspect conflicts can be identified. If required, other aspect conflicts may be identified and solved using the AORA conflict resolution tool [8].

In the best cases, establishing an order in which aspects are applied respecting their dependencies is enough. In the worst cases, a rethinking of which aspects should be applied or remodelled is required [8,29].

The composition task composes the aspectual scenarios with the base scenarios using the MATA language. The composition activity shows that the crosscutting concerns can be isolated in aspects and then composed into one or more base scenarios, without changing the application execution. By isolating the crosscutting concerns, we promote modularization, reuse, and the evolution capabilities of the application. Resulting models have tangled and scattered behavior because the base and aspectual scenarios were woven. During composition, new interactions and conflicts can be identified.

These should be used to refine and improve the information already contained in the catalogue.

Once the concern has been composed and validated for consistency, the last activity, development, starts. It consists on implementing both base scenarios for core concerns and aspectual scenarios for GIS concerns as independent modules. Finally, the two modules—core and GIS—are composed by a weaving engine, producing a final application that serves features of both concerns. In Sect. 10 we discuss some technological challenges.

So far we addressed a design where the base application concerns are oblivious with respect to candidate aspects and MATA properties have been introduced seamlessly in the approach.

Composition plays a primary role when a new requirement is needed in the application. For example, in a GIS application, a street segment may be blocked while a maintenance task is carried out, excluding the compromised segment in path-finding algorithm execution. This kind of unforeseen concern is handled using the composition step to handle its activation and deactivation: crosscutting relationships are specified using a crosscutting matrix. The solution is modeled using the MATA tool and composition is applied as required. Given that core concerns are oblivious with respect to the new unexpected GIS concern, this is easily introduced and removed from the application in the composition phase depending on the events that defines the volatile concern lifecycle.

In summary, let us stress a little more the creation of a catalog of GIS crosscutting concerns as a reuse mechanism. Once a crosscutting concern is identified, analyzed, and modeled, it is introduced into the catalogue for later instantiation. The catalogue will store, additionally to the models, concern usage information such as impact and results. Each time a concern is instantiated, it can suffer small refinements from user feedback, which will improve its representation and further re-use.

6 Case study

Let us illustrate our approach using the GIS Web application Maps@Web, aiming at helping users in their daily activities. This Web application provides a set of varied location-based services, including services related with cinemas, hotels, universities or police stations. For instance, if the user wants to go to the university, to the cinema and to the supermarket, all in the same evening, this application will calculate an appropriate path to visit all these places.

To better demonstrate the contributions of our approach, let us add the new requirement "Indoor Representation". This requirement changes the user's application context, taking him to an indoor representation of space when the building fits the whole map view. Next we show how to apply the three activities in our approach.

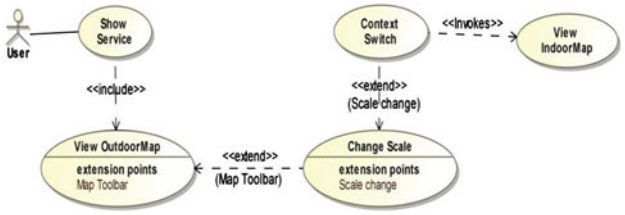

Fig. 8 Partial use case diagram for "Show Service". Use cases are enhanced with possible crosscutting concerns

Table 1 Relating crosscutting concerns with use cases

Use cases	CC concerns	
	Change scale	Context switch
Manage applications		
Create/edit main service	X	X
Manage service net	X	X
Search by service/address	X	X
Registration		
Edit profile		
Edit favourite places	X	X
Edit favourite categories		
Show suggestions	X	X

6.1 Identification of use cases and crosscutting relationships

Figure 8 shows a partial use case diagram, with the use case "Show Service" and the crosscutting concerns "Change Scale" and "Context Switch".

With the introduction of the "Indoor Representation" requirement, we now have two kinds of maps, Indoor Map and Outdoor Map. In this example, when the user starts using the application, an outdoor representation of the service location is shown, additionally exhibiting specific icons for points of interest. That is why the use case "Show Service" is connected with the concern "View Outdoor Map", with an ≪include≫ relationship. This last concern will be extended with "Change Scale" every time the map tool bar is changed. When the scale reaches the maximum zoom available, there will be a switch in the user's application context which enforces a swap between the outdoor and indoor views of the currently displayed location. We represent this relationship between "Change Scale" and "Context Switch" with an ≪<extend≫ stereotype. When the scale reaches the maximum and the context changes, the "View Indoor Map" is invoked.

Table 1 illustrates the crosscutting behavior between the requirements in the GIS concerns and other application

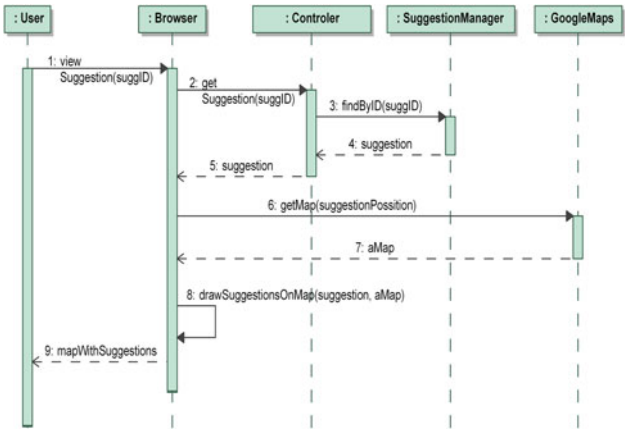

Fig. 9 Base scenario "Show service"

concerns, described by the corresponding use cases. As we can see, "Change Scale" and "Context Switch" are crosscutting concerns and they crosscut the same use cases; this may indicate that there is some interaction between them. This interaction will be solved later in this section.

6.2 Use case refinement and aspect modelling

The second activity of the approach specifies the base scenarios (Fig. 9) and the crosscutting concerns with sequence diagrams (Figs. 10, 11). Figure 9 illustrates the use case "Show Suggestion" from our example.

Figure 10 shows the crosscutting concern "Change Scale" represented with a MATA sequence diagram. The first message in this sequence diagram matches with the same one in the base scenario ("viewSuggestions(...)"), which is indicated by the ≪context≫ stereotype. Since changing scale is optional, it is represented with the "opt" fragment. Every message inside this fragment will be created in the base scenario, as is identified by the ≪create≫ stereotype.

Figure 11 shows the crosscutting concern "Context Switch" that will be activated by the concern "Change Scale". The first message of the sequence diagram in Fig. 11 matches the equivalent one in the base scenario. In this particular case, the base scenario is the aspect "Change Scale".

This sequence diagram shows that every time the user performs a change in the scale, the system will verify if a change in the spatial context is required. If this change is needed, the system accesses the information about the new context (in this case it is the indoor location). The "*any*" fragment allows the base scenario to continue its behavior, with a sequence of messages. At this point of the example, it is clear the interaction between "Change Scale" and "Context Switch". There are two indicating factors that point to this situation. First, in Table 1, these two aspects crosscut the same use cases. Second, the pattern matching of "Context Switch" will be made with another aspect, "Change Scale".

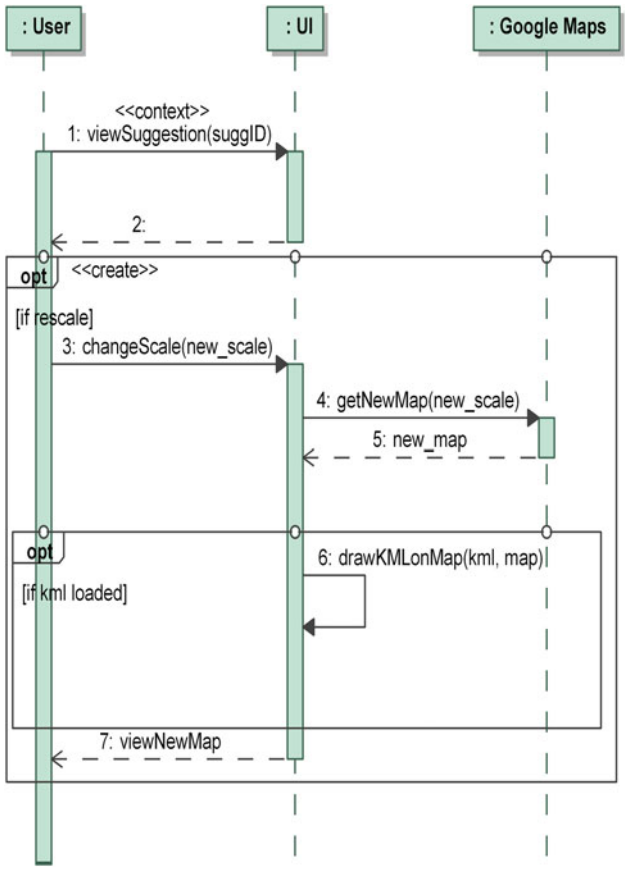

Fig. 10 Aspectual scenario "Change Scale"

Table 2 Aspect dependencies

	Context switch	Change scale
Context switch		Requires
Change scale	Provokes	

The relationships between these two aspects are shown in Table 2.

As the relationships in Table 2 indicate, for a spatial context switch, a change in scale is required, which is triggered by the user. Moreover, the use of the maximum zoom level will lead to a change in the user's spatial context. As we can see in this particular example, the relationships between aspects are simple, which means that it is enough to establish an order, taking into account their dependencies, in which aspects can be composed: "Change Scale before Context Switch".

6.3 Composition

This activity composes the aspectual scenarios with the base scenarios. Figure 12 depicts the base scenario "Show Suggestion" depicted in Fig. 9, composed with the aspectual

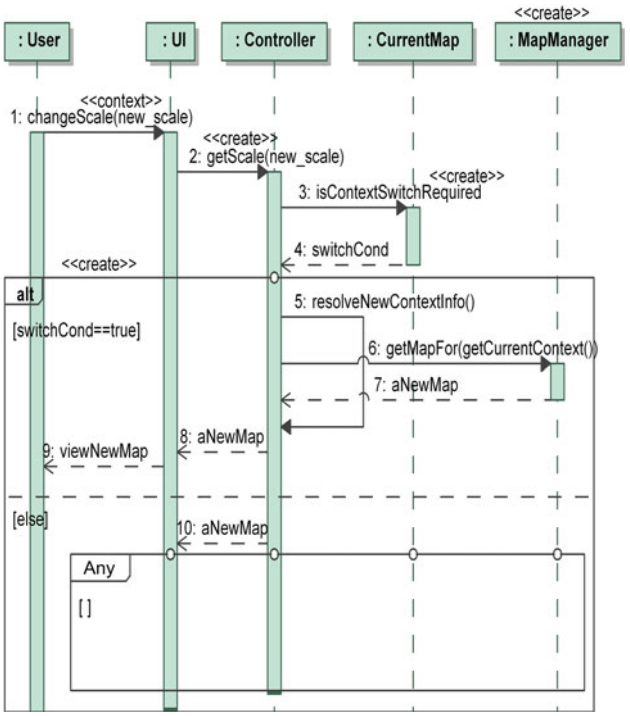

Fig. 11 Aspectual scenario "Context Switch"

scenarios "Change Scale" depicted in Fig. 10, and "Context Switch" depicted in Fig. 11. This composition is accomplished while respecting the MATA rules, defined as patterns in the aspectual scenarios and then, doing the pattern matching with the base scenario. As it can be seen in Fig. 9, the first message ("viewSuggestion") matches the first message of the diagram in Fig. 10. This matching represents a joinpoint where the aspectual scenario, in this case "Change Scale", will be inserted in the base scenario. The next pattern matching will be accomplished between the second message in Fig. 10 and the first message in Fig. 11. This means that the behaviour from aspectual scenario "Context Switch" is added to the composed scenario. The fragment "any" at the bottom of the aspectual scenario "Context Switch" will be matched with the rest of the aspectual scenario "Change Scale". Therefore, the matched fragment will be inserted in the composed scenario.

7 Building a knowledge base

So far we have characterized, analyzed and designed GIS crosscutting concerns promoting solutions that are characterized mainly by a weak coupling between resulting components, allowing a high level of reuse. Indeed, solutions must be both seamless, avoiding coupling between components and easing composition, and oblivious to the core concerns, removing any impact of new concerns introduction given that concerns are not aware to each other.

This was the drive to propose a reuse mechanism based on a catalogue of GIS concerns, defined in a very abstract and system-independent fashion. This catalogue should be used to help eliciting the problem domain concerns. The idea is to use these documented concerns when organizing the system space requirements (where the application domain is described), so that concerns of the problem domain are identified.

The knowledge base differs from GIS concern characterization (Sect. 4) in that the characterization helps detecting GIS concerns from an abstract point of view without addressing the underplaying application architecture and therefore its design. On the other hand, the knowledge base helps storing already designed concerns and any experience of its application, such as lessons learned.

The catalogue is fed with concerns as they are analyzed and designed by system architects. Nonetheless, these concerns are not static and can evolve as long as they are instantiated and reused in other applications being improved and, if required, modified, with usage feedback. Documented concerns are described with MATA models, which are used when this is recognized in the system space. These models are supposed to be refined as they are applied in different systems.

Each GIS concern can be documented using a simple template composed of the following five fields:

1. *Name* a short name describing the concern
2. *Description* a brief description of the concern;
3. *Requirements* concern's requirements must be clearly described for an appropriated understanding of current concern;
4. *Solutions* an explanation of how to solve the problem using the described aspect-oriented approach. Here the solution is documented using UML use cases and sequence diagrams combined with the MATA tool as it was explained in Sect. 5 and exemplified in Sect. 6.
5. *Experience* a brief description about the instantiation context in which this concern has been introduced. At least the application that initially required the GIS concern.

Furthermore, *known issues* and *consequences* items can aggregate value to the catalogue. The former will describe those exceptions and limitations in the appliance of the concern, while the latter will describe the impact in the target application.

7.1 Example

To illustrate the use of the catalogue, we next describe a concern, comprising a set of requirements, which enables the attachment of descriptive information to a location aware

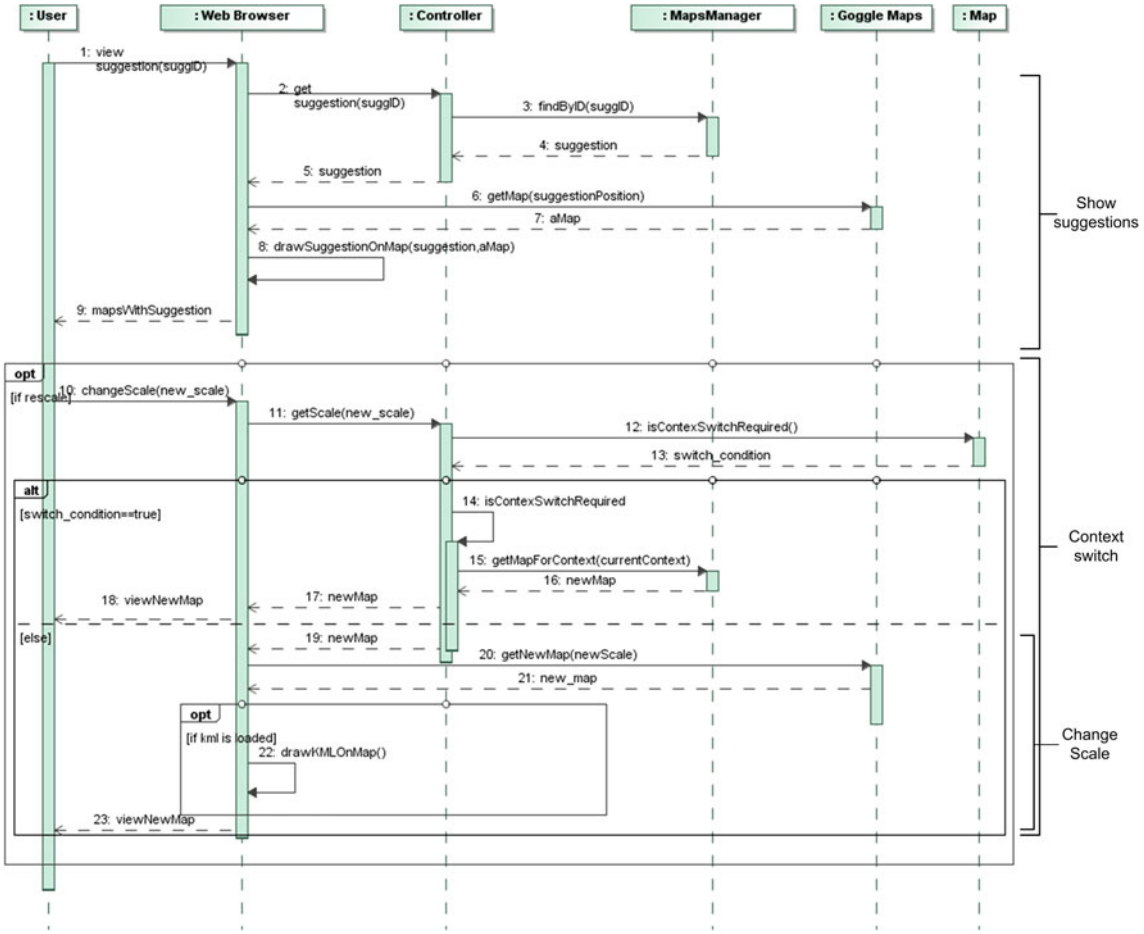

Fig. 12 Composed scenario

business object, such as a "Point Of Sales" or a "Hotel". A catalogue entry for this concern will look as the following:

- *Name* comment support
- *Description* a business object with spatial capabilities must be enriched with user comments information (the name of a place for example). It is commonly required to add new information to this data (e.g., to add a place description in a map).
- *Requirements* requirements for this concern are:

  Users can add comments to spatial object for sharing knowledge.
  Users need map-editing facilities for adding comments.
  Comments must be presented in a suitable fashion over the map.

- *Solution* The solution for this concern covers two models: class diagram and sequence diagram. Figure 13 shows a class diagram which introduces a relationship named "comments" that has, as source, any spatial object (object which is aware of its position) and, as target, a "Comment" class, holding a simple text variable. After applying the MATA composition process (described in Sect. 3.1) to the base business model with the MATA specification for the Comment concern, spatial business objects will contain a variable for a collection of Comments.

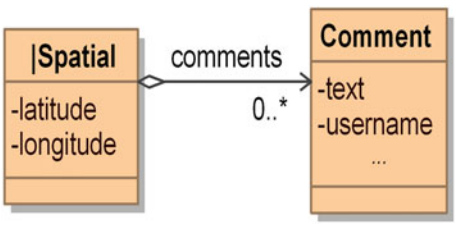

Fig. 13 MATA models for adding comments to business objects

To satisfy the user interface requirements, Fig. 14 presents a sequence diagram that decorates the visualization of Maps with the logic for rendering comments. That is, the diagram introduces a lookup sequence for determining

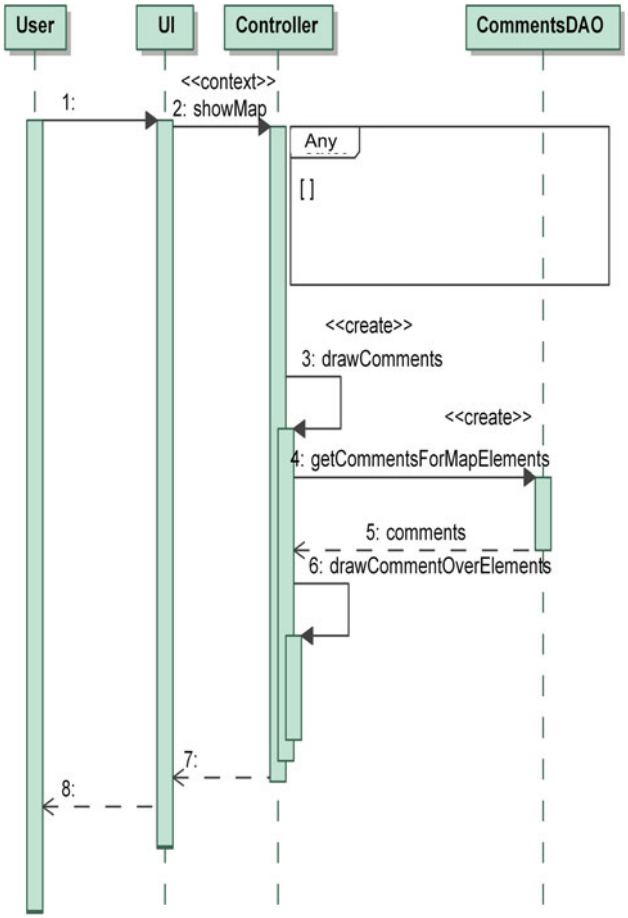

Fig. 14 Sequence diagram for presenting comments

Fig. 15 Comment concern example

registered comment for business elements placed into the current map's view, and a rendering sequence for drawing resolved comments.

- *Experience* This simple, but useful, aspect has been introduced successfully in applications, such as Google maps. The result was an improved user experience of the application, which follows a knowledge sharing alignment of Web 2.0 applications. Figure 15 shows the usage of the Comment concern.

8 Discussion

8.1 Modelling

The identification of crosscutting concerns during the early stages of the software development process has proved to be effective for improving modularization and thus increasing the localization software engineering principle, which facilitates maintenance of software applications. For example, we were able to modularize the concerns Context Switch, Change Scale, Comment support, which, otherwise, would be scattered in an object-oriented model, for example. Using the approach, these concerns were designed in an isolated fashion avoiding core concern being aware of them, simplifying the concern maintenance and reducing the system complexity.

In the context of Web-GIS applications, by identifying and characterizing spatial concerns early, according to their crosscutting nature (e.g. introducing scattered and tangled code into application's core concerns), we not only help improve spatial components' modularity but also improve their reuse. The catalogue of GIS concerns is the knowledge base where concern's information is stored. As it was shown in the example of Sect. 7, the "Comment support" concern can be instantiated in any application where, at least, one of its business objects matches the proposed MATA rule, that is, any application that has business objects with spatial information (latitude and longitude).

In this environment, the use of a tool like MATA simplifies the process of understanding how composition of separated concerns will work, and helps guaranteeing their correctness as well as some unwanted interactions.

The case study demonstrated how we were able to modularize the requirements "Change Scale" and "Context Switch" using aspects, which, otherwise, would be scattered along several other modules. By using the MATA tool, concern connections are expressed in terms of pattern matching expressions avoiding the replication of modeled behavior across the application. That is, everyplace a pattern is satisfied, the MATA model is instantiated.

8.2 Implementation

In this section we will describe briefly how above-mentioned ideas can be implemented using Java technology. We have developed a set of Web application prototypes that helped to assess the advantages and challenges of our current approach, in implementation tasks. Additionally, we also evaluated

prototype source code assessing well-known Object-oriented programming quality metrics.

8.2.1 Technical solution

In developed prototypes, Core and GIS concerns were developed having, as input, UML and MATA diagrams (Sect. 6) producing independent application modules. We used the Spring MVC Web framework [38] for implementing core modules. On the other hand, GIS concerns were developed asymmetrically using Java classes for business objects, AspectJ definitions for implementing crosscutting features specified in MATA models, HTML documents for user interfaces and, finally, XSLT documents for specifying user interface crosscutting features. We used the AspectJ tool for implementing AOP concepts such as introduction and method/s call interceptions in Core classes because it is the de facto aspect technology in the Java realm. XSLT transformations have been used in [17] successfully, for enriching HTML-based user interfaces with volatile functionality. By means of this kind of transformation, new widgets, layout changes and JavaScript code are introduced in HTML documents, crosscut by GIS concerns.

By using Maven [2], a Java-based build engine, modules were woven at compile time for producing an application that combines Core and GIS concerns. The Weaving process was an orchestrated execution of Maven's plugins, which comprised following tasks:

Compilation of core concern's and GIS concern's Java classes. Business objects and objects provided by GIS concerns are compiled;
Aspect weaving with Core Classes. Aspect realization of GIS concern are compiled and later woven with business objects that contain joinpoints;
Interface transformation using the XSLT engine. Because we use XML-based user interfaces specifications, XSLT transformations are applied over user interfaces that are affected by GIS features. This strategy can also be used for enriching any XML-based document such as configuration files;
Application packaging. Finally, the application shows original behavior enhanced with GIS.

Finally, the resulting Web application showed tangled GIS behavior among Core one. The application's user experience was enriched by a synergy concern relationship.

8.2.2 Source code quality analysis

In order to assess how our ideas impact on application implementation, we have analyzed different applications source code, measuring a variety of aspects producing well-known metrics of object-oriented programming [10]. In this analysis we will focus on the source line of code (SLOC or LOC) metric, which measures the size of the source code without comment lines; the lack of cohesion metric (LCOM) measure, where class' features are not related to its modularization; class complexity as the size of a class in terms of line of sentences; and code duplication as duplicated code sentences in several software artifacts.

We used the Sonar [37] source code quality tool for analyzing code automatically. This tool allows managing source code quality by analyzing code complexity, design, coding rules, duplications, potential bugs, among others. By using this tool, we compared how code changes when introducing volatile features using a conventional object-oriented approach (OOA) against using our proposed approach (AOA).

Before introducing the analysis results, we must remark that the lack of modularization of crosscutting concerns increases application complexity because the application is evolving and growing, affecting different application source code aspects. First we will analyze how using the OOA application reacts to new features and then we will provide a brief description of how AOA approach keeps modules simple.

The analysis showed that, when introducing volatile functionality using both conventional OOA and AOA, the SLOC metric increases. It is not surprising that the amount of lines of code registered as SLOC increases because new features are introduced and thus, new objects, and object's state and behavior is appended to object definitions. Even AOA-promoted modularization features must be implemented and thus its SLOC adds up to the overall amount. Although the SLOC metric may be used as an indicator for predicting defect density [34], AOA promotes the separation of concerns, keeping classes smaller and having different artifacts for each feature instead of having a single class with both core and spatial functionality tangled. This modularization has shown to prevent defects [14].

Application testability is compromised, at least, by two factors: the SLOC increment in an existing artifact, such as a class and the increase of the LCOM metric. When the new sentences are introduced in a class, its complexity increases, demanding new test cases for testing the new feature. On the other hand, the increase of lack of cohesion (LCOM) produces classes which encapsulate different features; this problem is known as the tyranny of the dominant decomposition [39], which does not modularize concerns that are not framed by the main decomposition criterion. In some cases, this issue was registered as a code duplication metric increase.

The consequence of volatile functionality elimination is an error prone task because it is of intrinsic crosscutting nature. Although we have not assessed the effort of manually removing a volatile functionality, several works have shown that the more changes a component has, the higher the risk for the

presence of bugs [19,30]. Instead, using AOA, there is no possibility of introducing a bug, because no code is modified when introducing a volatile functionality, thus a task for removing it is not needed.

9 Related work

The use of advanced separation of concerns techniques, and particularly the use of aspects, has been recently proposed for the development of complex Web applications, mainly to provide adaptation behaviours [6]. However, to our knowledge, there has been no research on the modularization of spatial concerns in Web-GIS software.

In the more general field of context-aware software, Munelly et al. [28] presents an approach for modularizing context-aware systems by encapsulating different types of context (e.g., location, user and device context) using an aspect-oriented approach. Our paper demonstrates that a context-aware application built in this way exhibits improved modularity, with corresponding improvements in comprehensibility, manageability and maintainability. Because it lacks of a process that gives support to the detection and design of adaptation concerns, this approach can be complemented with ours introducing the presented process for modelling spatial concerns in the early stages of software engineering.

Carton et al. [9] presents an approach to manage the development of applications for pervasive computing, based on a combination of aspect-oriented development techniques with model-driven development. This approach suggests modelling the pervasive application in Theme/UML and using model-driven transformations to gain the additional benefits of platform and technology independence. Besides the fact that this approach takes the advantage of model-driven development, our approach uses MATA to specify and compose aspects, and the process we presented is more elaborate than the one presented in Carton et al. [9].

Zipf and Merdes [44] discuss the use of aspects in GIS applications. However they only focus on the programming level while we focus on the earlier stages of software development. However, their analysis of possible spatial concerns is similar to ours.

Our approach can be enhanced with high-level aspect (HILA) [43] when an aspect introduces changes on components' state. HiLA presents a UML extension for modelling crosscutting behaviour in state machines that enables the graphical description of aspects' elements (like pointcuts and advices).

10 Conclusions and future work

Our approach modularizes spatial concerns (e.g., location-awareness and scale change), in Web-GIS applications. The process is composed of three main activities: identification, specification and composition. In this last activity MATA is used to compose aspectual models with base models.

We have shown that the quality of volatile applications like Web-GIS, always in constant change, can be improved by enhancing the respective modularity of spatial concerns. Our view represents a step forward with respect to existing approaches in which the spatial concerns are mixed with other crosscutting application concerns.

Currently we are working on the identification and modelling of additional spatial concerns with the aim of developing a catalogue, which will let us create new applications through composition, following the methodology presented in this paper.

Acknowledgments This research was partially developed within the project AspectWeb: Developing Web Applications with Aspects, funded by a bilateral agreement between Gabinete de Relações Internacionais da Ciência e do Ensino Superior de Portugal (GRICES) and Secretaria de Estado da Ciencia, Tecnologia e Inovação Produtiva da Argentina (SECyT), and partially funded by the ANPCYT of Mincyt, Argentina.

References

1. Alur D, Crupi J, Malks D (2003) Core J2EE patterns: best practices and design strategies. Prentice-Hall, Englewood Cliffs
2. Apache Maven. http://maven.apache.org/
3. Aronoff S (1989) Geographic information systems: a management perspective. WDL Publications, Ottawa. ISBN 0921804911
4. AspectJ. http://www.eclipse.org/aspectj/. Accessed 4 June 2011
5. Baniassad E, Siobhán C (2004) Theme: an approach for aspect-oriented analysis and design. In: 26th ICSE'04. IEEE Press, Scotland
6. Baumeister H, Knapp A, Koch N, Zhang G (2005) Modelling adaptivity with aspects. In: ICWE'05. LNCS, vol 3579. Springer, Berlin
7. Brito I, Moreira A (2003) Towards a composition process for aspect-oriented requirements. In: Workshop on early aspects 2003 at AOSD'03, USA, 2003
8. Brito I, Vieira F, Moreira A, Ribeiro R (2007) Handling conflicts in aspectual requirements compositions. Trans Asp Oriented Softw Dev Spec Issue Early Asp 4620:144–166
9. Carton A, Clarke S, Senart A, Cahill V (2007) Aspect-oriented model-driven development for mobile context-aware computing. In: 1st International workshop on software engineering for pervasive computing applications, systems, and environments at ICSE'07, USA, 2007
10. Chidamber SH, Kemerer CF (1994) A metrics suite for object oriented design. IEEE Trans Softw Eng (TSE) 20(6):476–493
11. Chitchyan R, Rashid A, Sawyer P, Garcia A, Alarcon MP, Bakker J, Tekinerdogan B, Clarke S, Jackson A (2005) Report synthesizing state-of-the-art in aspect-oriented requirements engineering, architectures and design. Lancaster University, Lancaster. AOSD-Europe Deliverable D11, AOSD-Europe-ULANC-9, pp 1–259
12. Coelho G (2008) Localização espacial de Serviços. M.Sc dissertation, Departamento de Informática, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Caparica, Portugal
13. Echo Web Framework. http://echo.nextapp.com/site/. Accessed 4 June 2011

14. El Emam K, Benlarbi S, Goel N, Melo WL, Lounis H, Rai SN (2002) The optimal class size for object-oriented software. IEEE Trans Softw Eng (TSE) 28(5):494–509
15. Filman R, Elrad T, Clarke S, Aksit M (2005) Aspect-oriented software development. Addison-Wesley, Reading
16. Gamma E, Helm R, Johnson R, Vlissides J (1995) Design patterns. Elements of reusable object-oriented software. Addison-Wesley, Reading
17. Ginzburg J, Rossi G, Urbieta M, Distante D (2007) Transparent interface composition in web applications. In: 7th International conference on web engineering (ICWE2007), Italy, July, pp 152–166
18. Google Web Toolkit. http://code.google.com/webtoolkit/. Accessed 4 June 2011
19. Hassan AE (2009) Predicting faults using the complexity of code changes. ICSE 2009:78–88
20. Heckel R, Küster JM, Taentzer G (2002) Confluence of typed attributed graph transformation systems. In: Graph transformation: first international conference, ICGT 2002. LNCS, vol 2505. Springer, Barcelona, pp 161–176
21. Jacobson I (2003) Use cases and aspects—working seamlessly together. J Object Technol 2(4):7–28. http://www.jot.fm/issues/issue_2003_07/column1.pdf
22. Jacobson I, Ng P (2005) Aspect-oriented software development with use cases. Addison-Wesley, Reading
23. Jayaraman P, Whittle J, Elkhodary AM, Gomaa H (2007) Model composition and feature interaction detection in product lines using critical pair analysis. In: MODELS international conference
24. Kulak D, Guiney E (2003) Use cases: requirements in context, 2nd edn. Addison-Wesley Professional, Reading
25. Laddad R (2003) AspectJ in action: practical aspect-oriented programming. Manning Publications, USA, ISBN 1930110936
26. Longley P, Goodchild M, Maguire D, Rhind D (2005) Geographical information systems and science. Wiley, New York
27. Moreira A, Rashid A, Araujo J (2005) Multi-dimensional separation of concerns in requirements engineering. In: 13th RE'05, pp 285–296
28. Munelly J, Fritsch S, Clarke S (2007) An aspect-oriented approach to the modularisation of context. In: 5th IEEE international conference on pervasive computing and communications, USA, 2007
29. Mussbacher G, Whittle J, Amyot D (2008) Towards a semantic-based aspect interaction detection. In: 1st International workshop on non-functional system properties in domain specific modelling languages at MODELS'08, France, 2008
30. Nagappan N, Ball T (2005) Use of relative code churn measures to predict system defect density. ICSE 2005:284–292
31. Oliveira A, Urbieta M, Araújo J, Rodrigues A, Moreira A, Gordillo SE, Rossi G (2010) Improving the quality of Web-GIS modularity using aspects. QUATIC 2010:132–141
32. OpenStreetMap. http://www.openstreetmap.org/. Accessed 4 June 2011
33. Peng ZR, Tsou MH (2003) Internet GIS: distributed geographic information services for the internet and wireless networks. Wiley, New York, ISBN 978-0-471-35923
34. Rahmani C, Khazanchi D (2010) A study on defect density of open source software. ACIS-ICIS 2010:679–683
35. Rashid A, Moreira A, Araújo J (2003) Modularisation and composition of aspectual requirements. In: 2nd AOSD'03, ACM
36. Rosenberg D, Stephens M (2007) Use case driven object modeling with UML: theory and practice. Apress, New York, ISBN 1590597745
37. Sonar. http://www.sonarsource.org/. Accessed 4 June 2011
38. Spring Framework. http://static.springsource.org/spring/docs/3.1.0.M2/spring-framework-reference/html/. Accessed 4 June 2011
39. Tarr P, Ossher H, Harrison W, Sutton SM (1999) N degrees of separation: multi-dimensional separation of concerns. In: 21st International conference on software engineering, Los Angeles, USA, May 1999. ACM, New York, pp 107–119
40. Whittle J, Jayaraman P (2007) Mata: a tool for aspect-oriented modelling based on graph transformations. In: Workshop on aspect-oriented modelling at MODELS'07
41. Whittle J, Moreira A, Araújo J, Jayaraman P, Elkhodary A, Rabbi R (2007) An expressive aspect composition language for UML state diagrams. In: ACM/IEEE MoDELS 2007. Lecture notes in computer science. Springer, Berlin, pp 514–528
42. Worboys M, Duckham M (2004) GIS : a computing perspective. CRC Press, Boca Raton
43. Zhang G, Hölzl M (2009) HiLA: high-level aspects for UML-state machines. In: Proceedings of the 14th Wsh. aspect-oriented modeling (AOM@MoDELS'09)
44. Zipf A, Merdes M (2003) Is aspect-orientation a new paradigm for GIS development? In: 6th Agile conference on geographic information science, Lyon, 2003