Using a Problem Domain Language to Specify Navigational Concerns in Web Applications

Leandro Antonelli[1], Silvia Gordillo[1], Gustavo Rossi[1], Joao Araujo[2], Ana Moreira[2]
[1]*LIFIA Facultad de Informatica, Universidad Nacional de La Plata, Argentina*
{lanto, gordillo, gustavo}@lifia.info.unlp.edu.ar
[2]*CITI/Dep. Informática, Faculdade de Ciências e Tecnologia*
Universidade Nova de Lisboa, Portugal
{ja, amm}@di.fct.unl.pt

Abstract

By nature, web applications involve a myriad of different concerns, which many times crosscut each other. The result is that these crosscutting concerns are scattered throughout different software artifacts provoking information tangling in those concerns. This paper presents an approach for using the problem domain language captured by LEL (Language Extended Lexicon) to improve the modeling of those concerns which affect navigation, i.e. the navigational concerns. It shows how to build partial navigation scenarios with user interaction diagrams, to analyze how they crosscut and, from there, how to obtain information for improving design models. Finally, it discusses how the interleaving of requirements elicitation with language specification allows improving the description of scenarios and the discovering of crosscutting relationships.

1. Introduction

Web applications are difficult to build as they usually combine the power of unstructured multimedia data and navigational access typical of hypertext systems, with other more conventional transactional behaviors. As web software usually allows multiple different user roles to interact with complex information systems, it is reasonable that many different and unrelated concerns arise. Many of them are idiosyncratic of web applications such as navigation, security and privacy; others relate to the specific problem domain. For example, in e-commerce we have recurrent concerns such as payment and advising; in e-learning we have evaluation. Some concerns are more general, like non-functional requirements (e.g. performance, and usability). The only way to deal with these concerns is to be able to correctly identify and modularize them, understand the impacts and trade-offs among them and the relationships between design artifacts that realize those concerns. Mature web engineering approaches have ignored these problems so far by providing only a limited set of mechanisms to improve separation of concerns. For example, object-oriented approaches like the OOHDM [15], UWE [9] or OOWS [14] use classes as the dominant decomposition technique. However, when a given concern does not fit the decomposition criteria, its realization is scattered along multiple decomposition artifacts (e.g. components, classes, nodes, methods), resulting in modules that are tangled. For example, security behavior (e.g. testing that the user has logged) is usually "tangled" with base functionality behavior and scattered among different modules.

Identifying concerns and encapsulating them in separate modules, independently from their crosscutting nature (functional or non-functional) is fundamental to support improved modularization, therefore facilitating reuse, traceability and evolution. Ideally, this should be considered since the early phases of software development, such as requirements analysis and architectural design. In this regard, the Early Aspects community has been proposing mechanisms to deal with the identification, representation, and composition of aspects at requirements level [6].

In this paper we propose an approach for dealing with navigational concerns, i.e. those requirements which affect navigation in web applications. Particularly, we show how to use the Language Extended Lexicon (LEL) [10] both to improve understanding of navigation concerns and to identify concern crosscutting. The contributions of this paper are three fold: (i) describe an aspect-oriented requirements approach to handle navigational concerns, characterizing the situations in which crosscutting exists; (ii) show how to use the LEL to

improve the requirements modeling process and (iii) show how to improve the structure of LEL to handle multiple concerns.

The rest of the paper is structured as follows. Section 2 motivates the work. Section 3 presents some background needed to support our approach. Section 4 introduces navigational concerns. Section 5 gives an overview of the approach. Section 6 applies the approach to a case study. Section 7 discusses some related work. Finally, Section 8 draws some conclusions and points out directions for future work.

2. Motivating example

Throughout this paper we will use as an example a web-based application for management of trials related with tax evasion. In the state of Buenos Aires, when a person (or company) fails to pay corresponding taxes, an executive bond is issued with a date limit. After this date, the prosecutor's office can initiate a trial which evolves according to different criteria (taking into account the nature of the debt, the owner, the existence of a moratorium, etc).

Figure 1 shows a snapshot of a web page containing information on a trial which has already been initiated. Trials have a normal flow consisting of: initiate, intimating or warning the debtor, inhibit his belongings and eventually seize him/her or even send him to jail. There is an alternative flow when a moratorium exists and the debtor is included in it. This application assists the prosecutor in progressing through the trial, allowing him to find information on related trials, advising him on how to proceed according to the current state, etc. Finally, the application is personalized providing access rights to different levels of information about trials.

In Figure 1, information, relationships and operations corresponding to different (and even unrelated) concerns are present: the *"trial information"* concern, marked with ellipse 1, allowing to obtain the basic information and relationships; the *"trial prosecution"* concern at the right for letting prosecute the trial (2); the *"assistance"* concern allowing to suggest about how to prosecute the trial (3); the *"personalization"* concern depicted in (4). Not evident in the interface, but playing a role in the "shadows", is the *moratorium* concern which somehow constrains information and available operations. For example, if there is an open moratorium and the debtor decides to enter in the moratorium, it might happen that the trial cannot proceed.

Notice that these concerns might be realized or supported by different kinds of software artifacts, according to their relationships, eventual crosscutting, etc. Besides, the fact that they show up together in a single page is the consequence of a critical design decision (at the navigational design level) and not just an interface issue. The rationale behind these design decisions should be clearly recorded and traced.

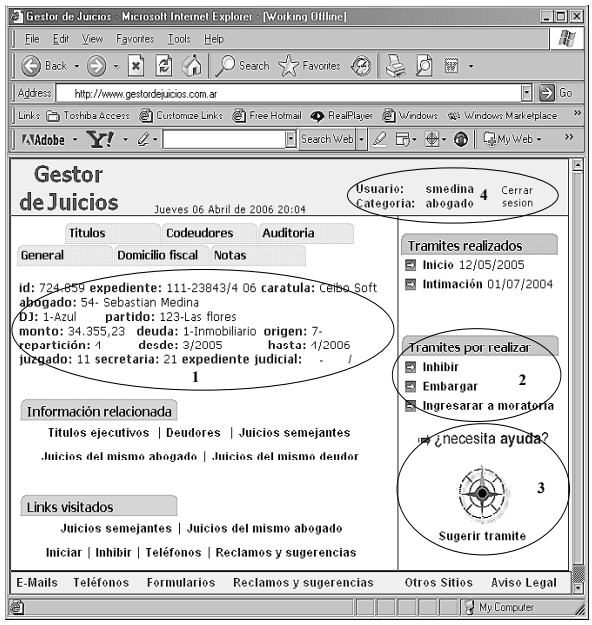

Figure 1. Different concerns (marked with ellipses) in the same page

3. Background

This paper builds on work already developed for requirement engineering [1, 10, 11] and modeling of web applications [8]. The following two subsections introduce some basic principles on both areas.

3.1 Language Extended Lexicon

The Language Extended Lexicon (LEL) is a glossary which captures the problem domain language [10]. It ties to a simple idea: "understand the problem language without worrying about understanding the problem". Its principal goal is to record important symbols also known as entries (i.e., words or phrases) of the domain. Each LEL's symbol has two types of descriptions: the notion, which is the meaning of the symbol; and the behavioural responses (or impacts), which describes how the symbol influences others or how other symbols impact on it. Notion and behavioural responses can be described using several sentences. These sentences should include mostly other LEL entries (this is call circularity principle);

words “external” to the LEL should have a precise meaning to allow that the language is defined in terms of itself. As an organizing principle, LEL symbols are classified into four categories: subjects, objects, verbs and states. Table 1 shows a schematic template with the guidelines we use to represent notions and impacts of each category.

Table 1. Categorization of LEL symbols and descriptions of each one

Categories	Description	Notion	Behavioural responses
Subject	Active domain entity which perform actions	Which information it manages	Actions it performs
Object	Elements manipulated by subjects	How it is composed	Actions performed on it
Verb (action)	Identifies tasks that subjects perform on objects	Goal	Sub-Goals
State	Situations of subjects and objects as the result of actions.	What it represents	Previous and subsequent states

In this paper we specify LEL entries using an XML representation. Examples can be found in Figures 2-5, which illustrate the LEL specifications for Trial (an object), Personalize (verb), Prosecute (a verb), and Moratorium (an object), respectively. The <Ref> tag indicates relationships among symbols that can be explored with a hypertext tool [1].

```
- <Symol="Trial">
    <Category>Object</Category>
    <Notion id=”1”> A Process to oblige a <Ref id=”debtor”
        >debtor</Ref> to either pay his debts or being
        punished by justice </Notion>
    <Notion id=”2”> A trial is identified by its cover, it is carried
        out by a lawyer against one or more <Ref id=”debtor”
        >debtors</Ref> </Notion>
    <Notion id=”3”> A trial is based on one or more <Ref
        id=”executive document” > executive documents
        </Ref> which formalize the debt</Notion>
    <Notion id=”4”> A trial is composed of a sequence of steps
        </Notion>
    <Behavioural response id=”1”>It can <Ref id=”prosecute”
        >proceed in a normal way</Ref> or different actions
        are performed if the debt is in a <Ref
        id=”moratorium” >moratorium</Ref> </Behavioural
        response>
</Symbol>
```

Figure 2. Trial symbol in the LEL

```
- <Symol="Personalize">
    <Category>Verb</Category>
    <Notion id=”1”> The application adapts the information and
        actions according to the use role </Notion>
    <Behavioural response id=”1”> The application adapts itself
        to lawyers, head of lawyers, judges and debtors
        </Behavioural response>
  </Symbol>
```

Figure 3. Personalize symbol

```
- <Symol="Prosecute">
    <Category>Verb</Category>
    <Notion id=”1”> A set of steps to be followed in a normal
        trial to oblige the debtor to pay </Notion>
    <Behavioural response id=”1”> <Ref id=”initiate”
        >initiate</Ref> </Behavioural response>
    <Behavioural response id=”2”> <Ref id=”warn” >warn</Ref>
        </Behavioural response>
    <Behavioural response id=”3”> <Ref id=”inhibit”
        >inhibit</Ref> </Behavioural response>
    <Behavioural response id=”4”>if the debtor has properties,
        <Ref id=”seize”>seize</Ref> them </Behavioural
        response>
  </Symbol>
```

Figure 4. Prosecute symbol

```
- <Symol="Moratorium">
    <Category>Object</Category>
    <Notion id=”1”> Opportunity to cancel an old debt, paying it
        in installments </Notion>
    <Behavioural response id=”1”>A lawyer cannot make the
        <Ref id=”prosecute”>trial</Ref> progress until the
        moratorium ends. </Behavioural response>
    <Behavioural response id=”2”> A <Ref
        id=”debtor”>debtor</Ref>> is obliged to pay his
        monthly installments </Behavioural response>
    <Behavioural response id=”3”> The judge cannot use the case
        as a reference for other trials </Behavioural response>
  </Symbol>
```

Figure 5. Moratorium symbol

3.2 User Interaction Diagrams (UIDs)

In this paper, we use User Interaction Diagrams (UID) [8] to improve the specification of use cases by describing partial navigation scenarios. A UID represents an interaction between the user and a system to fulfill a task. It describes the exchange of information with a simple state machine-like notation, wherein each interaction step is represented as an ellipse and transitions between interaction points as arcs. In Figure 6, we show a UID for the use case “finding a trial given its cover”. In the first interaction of Figure 6 (indicated by an incoming arrow), the user has to enter the trial’s cover (or part of it), denoted by a rectangle. The system’s response is either one trial or a set (represented by “…”) of trials, matching the user input. For each trial some related information (e.g. cover, lawyer) is shown. The user chooses one of them (represented by “1”), and the system returns the complete information on the trial (represented inside the third ellipse). Additionally, the user can perform

several operations, (e.g. show executive documents, show debtors, show -administrative- steps) indicated by a line with a black bullet. The complete syntax for UIDs can be found in [8]. UIDs provide a first insight about the information and linking structures that the application will manipulate, and an outline of the possible (partial) navigation flows that can be validated with stakeholders. Using simple heuristics (described in [8]) conceptual and navigational models can be derived from UIDs.

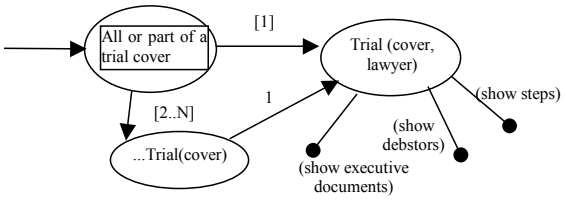

Figure 6. A simple UID for "finding a trial given its cover"

4. Navigational Concerns

According to [13] a concern implies any coherent set of requirements with interest for one or more stakeholders of the problem domain, e.g. all requirements referring to a particular theme or behavioral application feature. We say that a concern is navigational when some of its requirements affect the application's navigational structure, i.e. the way users navigate throughout the application. In other words, navigational concerns comprise those requirements which in a direct or indirect way affect the contents and linking options of a web page. As an example, *Assistance* is a navigational concern which comprises those requirements that indicate how the system can advise the lawyer to proceed with a trial; meanwhile, *Persistence* is an important application concern which should have no impact on navigation.

Navigational requirements are important for several reasons. First, because it is well-known that the quality of web software depends on the navigation facilities provided to support different users' tasks. Besides, by focusing on navigational concerns we can improve the specification of core design artifacts, i.e. those which are subject of exploration and which are affected by users' actions.

We extend the usual notion of crosscutting concern and say that navigational concerns crosscut when the realization of one of them affects others, or when they manifest together, for example, in a web page. Crosscutting concerns may affect other concerns; they may appear as information or behavior that does not seem to strongly belong to the current node, or as links that "break" the current task flow. The concerns shown in Figure 1 are examples of possible crosscutting navigational concerns. Crosscutting in navigational concerns may be originated in the intended interaction and navigation facilities, e.g. we want the lawyer to be able to learn about other trials while prosecuting one, or we may want him to receive assistance or advice before proceeding. We also want to keep a record of the lawyer navigation history and keep it permanently available, etc. Therefore, identifying crosscutting concerns also helps to find non obvious links or operations.

In this paper, we argue that crosscutting of navigational concerns can be identified early during requirements specification by analyzing the LEL and the information in UIDs.

5. Our approach in a nutshell

Our complete model, illustrated in Figure 7, has been inspired by [13]. In this paper we will concentrate on concern identification and specification activities. More details about composition of crosscutting concerns can be read in [13] and [2].

We begin by instantiating a catalogue of meta-concerns, those concerns which appear time and again during system development. Meta-concerns are described in a very abstract and system-independent fashion and are used to help eliciting the problem domain concerns. Specific concerns can be seen as specializations of concerns at the meta-space level. Meta-concern and system spaces are represented by ovals in Figure 7.

Both the abstract concerns in the meta-concern space as well as their concrete realizations will be defined using a set of well-defined templates based on XML as described in [13]. Meta-concerns can be generic or domain specific. In this paper we will focus mainly on navigational meta-concerns such as Navigation History, Advising, Personalization, etc.

We use the LEL to support the meta-concern instantiation process and to enrich the specification of requirements in a concern. As in [12] the elicitation process is made incrementally, so the used terms are incorporated in LEL in parallel to the requirements specification; both tasks (specification and LEL construction) enrich each other.

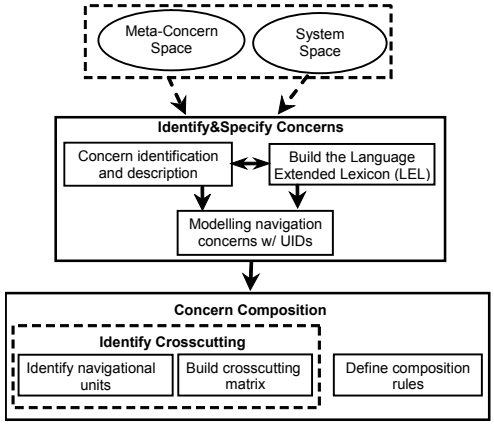

Figure 7. Overview of the approach

LEL will be used to complement concerns descriptions (and respective UIDs, described later). The LEL is built in two stages. First, we build and validate one LEL for each concern (particularly for navigational concerns as shown in Table 2). The same term might appear in different concerns, perhaps with slightly different meanings, or a description in one concern might use symbols defined in other concerns. By analyzing how terms appear in different concerns and how definitions contain information defined in other concerns we get insight about possible crosscutting.

Table 2. Symbols in each concern

Concern	LEL symbols
trial information	trial, step, debtor, executive bond, stat
prosecute	initiate, intimate, inhibit, seize, get into moratorium
moratorium	moratorium, pay his monthly installments, get out of moratorium, get into moratorium
personalization	personalization, lawyer, head of lawyer, judge, debtor
assistance	assist

The identification of the concrete concerns is achieved iteratively and incrementally, by handling small parts of the problem domain at a time. New meta-concerns might appear by generalizing new application concerns.

Once the description of the problem domain has been organized into concrete specializations of concerns from the meta-concern space, we can model the interactions related to each concern. To do so, the requirements in each navigational concern that involve user interactions are modeled using a UID as described in Section 3.2. Some navigational concerns, such as authentication, or moratorium, will encompass requirements that do not have an associated user interaction, e.g. by implying a system check against the user identity (authentication), and some pre-processing before an operation is performed (moratorium). In this case, there will be no associated UIDs, though they are also analyzed regarding crosscutting. Composition of concerns is then realized at the UID level and is achieved in two steps: (1) identify navigational units and build a crosscutting matrix between navigational units and navigational concerns; (2) build a composition rule for navigational units. Composition will be used, for example, to specify and analyze possible navigation paths separately. The advantage is that navigation models can be changed with a localized impact. Though it is not the focus of the paper to fully describe the composition process, we summarize it below.

A navigational unit (NU) is the requirement counterpart of a node in the navigational model and reflects an information structure that emerges from an interaction state in a UID (e.g. a trial, an executive bond, etc). NUs usually represent some data items presented to the user, and are generally the basis for performing the next interaction state in a UID, e.g. because some information is selected. Most concerns encompass at least one NU though it might be not necessary to define one NU per concern. Some requirements might give raise to more than one NU, such as the sequence of information structures arising during a trial update.

Crosscutting is represented in a matrix of the form *Navigational Unit* X *Concern* (not shown in this paper due to lack of space). For each navigational unit we analyze if it needs to provide information or operations pertaining to other navigational concerns, i.e. if the navigational concerns are accessible from or impact in that navigational unit. This situation is marked with a tick in the corresponding cell. Likewise, for each column (concern) we mark if this concern influences the corresponding navigational unit. For example, some concerns might not involve a specific NU (e.g. personalization), but they dictate rules about how a NU should look like, for example which information should be shown according to the user role.

A crosscutting concern is defined as a concern that is scattered among several navigation units. This means that the navigation units are tangled, since they include the properties of the dominant concern that originated them as well as the properties of this new navigation concern. The LEL is used to identify crosscutting concerns by analyzing the impact of symbols in each concern with respect to symbols in

others. When a NU is itself a symbol in the LEL (e.g. Trial), the analysis is straightforward.

The second step is to define composition rules that will show the impact of crosscutting concerns on navigational units. For those concerns in which some use cases are described in terms of UIDs, composition rules express specific compositions between UIDs. In some cases the identification of crosscutting navigational concerns will not imply the composition of UIDs but meaningful information to be used in subsequent development activities (e.g. for crosscutting involving concerns not represented by UIDs such as moratorium).

The identification of crosscutting navigational concerns is essential to better trace the rationale of architecture and design decisions including the definition of an adaptable navigation model or the possible use of aspects down in the development lifecycle.

6. CASE STUDY

6.1 Identify and describe concerns

From the description of the application case study, we can identify that several meta-concerns can be reused here. For example, *Information Retrieval* to access the relevant information of the system (e.g. trials), *Information update* to register the result of each step in the trial, *Personalization* to customize the application to a specific user and *Assistance* to provide guidance about how to prosecute a trial. Figure 8 shows the abstract definition of the *Information Retrieval* meta-concern and Figure 9 one concrete instantiation, describing the *Trial Information* concern. The *Trial Prosecution* concern might be considered an instantiation of *Information Update* (not shown here due to lack of space).

```
<MetaConcern name="InformationRetrieval">
    <Description>The operation of accessing information from
        a computer system </Description>
        <Examples>Database retrieval, Multimedia
        retrieval</Examples>
    <Relationships> Availability, Mobility, InformationUpdate
        </Relationships>
</MetaConcern>
```

Figure 8. The *InformationRetrieval* meta-concern

```
<Concern name="Trial information">
  - <Requirement id="1">
        The system will be accessed to provide basic
        information about the trial
  </Requirement>
  </Concern>
```

Figure 9. The *Trial Information* concern

Figures 10 and 11 show a snapshot of some other concrete concerns.

```
- <Concern name="Moratorium">
   - <Requirement id="1">
        If the debtor enters into a moratorium, the trial must be
        suspended.
   </Requirement>
   - <Requirement id="2">
        If the debtor does not pay the moratorium, the lawyer
        must prosecute the trial again.
   </Requirement>
</Concern>
```

Figure 10. *Moratorium* concern

```
- <Concern name="Assistance">
  - <Requirement id="1">
        The lawyer is recommended how to proceed a trial
        according to its current state
  </Requirement>
  - <Requirement id="2">
        The debtor is recommended to enter a moratorium if
        there is one
  </Requirement>
  </Concern>
```

Figure 11. *Assistance* concern

6.2 Build UIDs from LEL and Concerns Descriptions

We use the LEL as a guide to specify the UIDs in the same way as the LEL has been used to describe use cases [5] or scenarios [11]. We next show two simple guidelines which can be used:

- LEL entries classified as objects (respectively subjects) include in their *notion* meaningful insight to determine which information is presented to the user. When this information is itself composed of other objects, we can find a possible navigation step (Figure 12).
- LEL entries classified as verbs contain in their impact how the action is de-composed in sub-actions; we can use this information to describe the interactions for use cases involving a verb (action) (Figure 13).

Figure 12 shows the complete UID corresponding to the use case: “Show the information of a trial given its cover”, corresponding to the *Trial Information* concern. Trial is an object which has a LEL entry (Figure 2); it indicates that a trial is identified by a cover. The notion indicates information which is relevant for a trial: the lawyer, debtors, bonds, steps. As debtors, bonds and steps are defined as LEL entries (not described in the paper for conciseness).

We can build a navigation sequence to get information on them as shown in Figure 12.

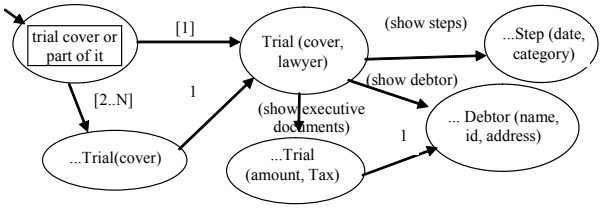

Figure 12. Complete UID for showing a trial given its cover

In Figure 13 we show the UID: "Initiate a trial from an executive bond". One of the trial's impact (in the LEL) is that a trial is initiated from a debt document. Meanwhile, initiate trial is a verb in LEL; its impacts show what it means to initiate a trial, namely indicating date, court and eventually secretariat. Corresponding bubbles are therefore shown in Figure 13.

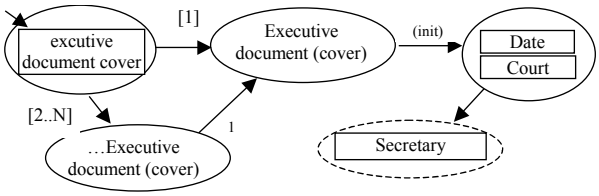

Figure 13. UID for initiating a Trial

6.3 Identify crosscutting

The identification of crosscutting navigational concerns provides interesting information about the system to be developed according to the nature of the crosscutting. For example, as shown in Figure 1, some crosscutting might "just" indicate that in the same web page we will include information and operations corresponding to different concerns; this fact has an important consequence when defining the linking structure of the application as it gives us information about which links are necessary and also about the structure of the corresponding pages (e.g. which information must be shown). Other concerns might give raise to more complex architectural decisions, such as the use of aspects.

Analyzing requirements such as those in Figures 8-11 and symbols in the LEL for each concern as in Figures 3-6 we got the following information about crosscutting. The concern *Moratorium*, which comprises requirements related with a different way of treating debtors, crosscuts operations related with a trial, *Trial Prosecution* (as the order of steps changes, or even some steps can be undone) and also *Trial Information* as it might constraint which information is to be shown.

Personalization (Figure 3) also crosscuts *Trial Information* and *Prosecution* as it indicates which information each type of user can see, and which operations are available. *Assistance* also crosscuts *Personalization* as the kind of assistance depends on the role of user. It also crosscuts *Prosecution* as it might also involve which options for prosecution are available.

As the LEL has been organized in concerns, by analyzing the impacts of a symbol on others in different concerns we can also get information about crosscutting. For example in the *Assistance* concern the most meaningful symbol is "Assist". In the impacts of the symbol each type of user of the system is referenced together with the actions they can perform; these types of users have been defined in the concern *Personalization*: lawyer, head of lawyers, judge and debtor; analogously the actions described in Assistance have been defined in the *Prosecution* concern.

By combining the information obtained from the analysis of UIDs with this analysis on the symbols of the domain language we can better understand which crosscutings we have to deal with, and their essence.

7. Related Work

Several Aspect-Oriented Requirements Engineering (AORE) approaches have appeared lately, such as [4] and [13]. These are classified as symmetric and are closer to the work in this paper. In [4] the Theme approach supports the requirements analysis activity by providing a mechanism to identify base and crosscutting behaviors from a set of actions. An action is a potential *theme*, which is a collection of structures and behaviors that represent one feature. The results of analysis are mapped to UML models. In [13] Moreira et al. define a multi-dimensional approach to separation of concerns in requirements engineering as well as trade-off analysis of the requirements specification. One of the key elements of the approach is the notion of a meta-concern space, a catalogue of typical concerns, functional and non-functional, that manifest themselves time and again in various software systems. The abstract concern definitions in this meta-concern space are used as a basis to delineate requirements into concrete concerns. These definitions are adopted in this work.

Nevertheless, none of these approaches address navigational concerns explicitly. Given the specificity

of navigational concerns, there is a need to tackle both the specification of such concerns, as well as their respective compositions. Here, we adapted a well known navigational modeling technique to support the specification and composition of crosscutting navigational concerns. The idea of navigational unit has been inspired in the concept of navigation semantic unit in [3]. In their approach for navigation analysis, the authors derive semantic units and navigation semantic links from use cases. Similarly, our navigation units are derived from UIDs (a graphical representation of use cases) and also represent a first step in the definition of navigation classes (nodes and links). Our approach can be used to improve navigation analysis by introducing concerns and crosscutting concerns. In [11] Leite et al. propose a strategy to obtain domain scenarios from the LEL, particularly from verb entries. Instead we derive application UIDs by also using subjects and objects to elicit meaningful information structures which then evolve into navigation units and later into application classes. We also use the LEL to identify crosscutting concerns.

8. Concluding Remarks and Future Work

In this paper we have described an approach for specifying navigational concerns in web applications at requirements analysis level. The approach combines the use of the Language Extended Lexicon with modern early aspect techniques to model and later compose navigational concerns. We have shown with some examples how we describe a concern and its requirements and how we use LEL to help us describing partial navigational scenarios with User Interaction Diagrams (UIDs).

We are currently working on a composition approach for crosscutting navigational concerns by analyzing only navigational units [7]. Composed UIDs may help stakeholders to determine different design trade offs. For example, by composing *Trial information* with *Assistance* (as shown in Figure 1) we can improve usability without significant cognitive overhead (e.g. as a consequence of interface complexity). Information gathered from crosscutting navigational concerns can help to improve existing guidelines (e.g. [8]) to obtain design artifacts. Analyzing the nature of crosscutting navigational concerns gives us a better insight to decide which modularization mechanisms apply for those design artifacts (e.g. when we use an object-oriented approach). By analyzing these simple partial navigation scenarios and their compositions we gather important information to be used during design. This information includes:

- which links are necessary to support navigation, which implies what relationships application objects must support;
- what information and operations, application objects must comprise; when this information and operations are available (e.g. always, according to the concern in which the object was accessed, etc.).

When the same class provides sets of independent behaviors (e.g. belonging to different concerns) we can simply juxtapose those behaviors in the same class. For example, we can juxtapose the behavior of a trial that supports methods for calculating relationships for the *Assistance* concern with behaviors that allow dealing the Trial in a *Moratorium*. We can also evaluate to maintain those behaviors separated in aspects.

We have started building a prototype tool for the IBM Eclipse to support a meta-concern repository, to describe UIDs and LEL entries and to use the gathered information to generate application templates development tool.

References

[1] L. Antonelli, "Traceability in the requirements elicitation and specification", Master Thesis (in Spanish), UNLP, Facultad de Informática, 2003.

[2] J. Araújo, J. Whittle, and D. Kim, "Modeling and Composing and Validating Scenario-Based Requirements with Aspects", *in proceedings of the 12th International Requirements Engineering Conference*, Kyoto, Japan, 2004.

[3] C. Cachero, N. Koch, "Conceptual Navigation Analysis: a device and platform independent navigation specification", *2nd International Workshop on Web Oriented Software Technology*, Málaga, Spain, 2002.

[4] Clarke, S., E. Baniassad, *Aspect-Oriented Analysis and Design. The Theme Approach.* Addison-Wesley, Object Technology Series, ISBN: 0-321-24674-8, 2005.

[5] L.M. Cysneiros, J.C.S.P. Leite, "Using UML to Reflect Non-Functional Requirements", *In proceedings of the 11 CASCON*, IBM, Canada, Toronto, 2001, pp 202-216.

[6] Early Aspects Home Page. In www.early_aspects.net.

[7] S. Gordillo, G. Rossi, J. Araujo, A. Moreira, "Identifying and Composing Navigational Concerns in Web Applications Requirements". Submitted

[8] N. Güell, D. Schwabe, P. Vilain, "Modeling Interactions and Navigation in Web Applications", *ER (Workshops),* Utah, USA, 2000, pp 115-127.

[9] N. Koch, A. Kraus, R. Hennicker, "The Authoring Process of UML-based Web Engineering Approach", *In Proceedings of the 1st International Workshop on Web-Oriented Software Construction (IWWOST 01),* Valencia, Spain, 2001, pp 105-119.

[10] J.C.S.P. Leite, A.P.M. Franco, "A Strategy for Conceptual Model Acquisition", *In Proceedings of the First IEEE International Symposium on Requirements Engineering*, San Diego, California, IEEE Computer Society Press, 1993, pp 243-246.

[11] J. C. S. Leite, G. Hadad, J. H. Doorn, G. Kaplan, "A scenario construction process", *Requirements Engineering 2000*, Springer-Verlag London Limited , 2000, pp 38-61.

[12] Loucopoulos P., V. Karakostas, *Software Requirements Engineering*, McGraw- Hill , 1995.

[13] A. Moreira, A. Rashid, J. Araújo, "Multi-Dimensional Separation of Concerns in Requirements Engineering", *in Proceedings of the 13th IEEE International Requirements Engineering Conference (RE 2005),* IEEE Computer Society, Paris, France, 2005.

[14] O. Pastor, S.M. Abrahão, J. Fons, "An Object-Oriented Approach to Automate Web Applications Development", *in Proceedings of EC-Web 2001*, Munich, Germany, 2001, pp 16-28.

[15] D. Schwabe, G. Rossi, "An Object-Oriented Approach to Web-Based Application Design", *Theory and Practice of Object Systems (TAPOS)*, Vol 4, 1998, pp 207-225.