STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 3, 2013

A MULTI-DIMENSIONAL SEPARATION OF CONCERNS OF THE WEB APPLICATION REQUIREMENTS

CALIN-EUGEN-NICOLAE GAL-CHIS[(1)]

Abstract. The implementation phase of the requirements in web applications is depending on a certain level by the platform of deployment, the architectural choice, the developer expertise and experience, is depending on the programming languages, and of the framework and tools selected to be used during the development process. A methodology of sorting and grouping web application requirements is needed for two reasons: the first is to make sure that the requirement fits a specific form of description in the recording of own characteristics and second, to assign the requirements to the appropriate deployment method, to a certain type of developer in his domain of expertise. An approach based on multi-dimensional separation of concerns is proposed to guide the process of implementing the requirements.

keywords: requirements engineering, web applications, separation of concern, software architecture

1. Introduction

In todays web applications development, requirements engineering (RE) is making important steps towards formalizing the requirements. Requirements are supposed to be implemented into the application. In terms of implementing the requirements the only category of people interacting with the requirements are the developers. By developers we do not understand just programmers, but include here web designers, software architects, testers, tools of deployment or any other actor interacting direct or indirect with the code of the developed product, with the application data or with the system configuration of the

Received by the editors: March 30, 2013.

1998 *CR Categories and Descriptors.* D.2.1 [**SOFTWARE ENGINEERING**]: Requirements/Specifications – *methodologies*; D.2.9 [**SOFTWARE ENGINEERING**]: Management – *Life cycle*.

Key words and phrases. requirements engineering, web applications, separation of concern, software development.

This paper has been presented at the International Conference KEPT2013: Knowledge Engineering Principles and Techniques, organized by Babeş-Bolyai University, Cluj-Napoca, July 5-7 2013.

softwares environment. In the process of implementing one requirement, the developers are affecting at least one of the components of the MVC (Model-View-Controller) architectural pattern in at least one aspect.

In web applications and not only, the software architecture [2] is the process of defining a structured solution that meets all of the technical and operational requirements, while optimizing common quality attributes such as performance, security, and manageability. It involves a series of decisions based on a wide range of factors, and each of these decisions can have considerable impact on the quality, performance, maintainability, and overall success of the application.

The software architecture is focused to organize functionalities in areas of concern, such as data layer, busyness layer, service layer, presentation layer and other connected systems. The choice of architecture solutions in web applications is vast, a selection of the solution depending of several factors, such as ease of deployment, reduced cost, ease of development, reusability, mitigation of technical complexity.

The RE is providing methodologies to define the requirements. Once they are defined, requirements have to be implemented into the application, as part of the requirements lifecycle. This paper introduces an approach for mapping the requirements to the software architecture by using a separation of concerns based methodology.

2. Related work

Even though one of the uses of RE is exploratory, such as while specifying requirements, the problem to be solved is better understood; the goal of RE in the Software Engineering is to obtain a stable set of requirements, which serves as basis for the further steps in the development process. According to Lowe and Hall, three activities are used to achieve this goal: elicitation, specification, and validation of requirements [1].

The elicitation of requirements is the activity by means of which the functionalities of the system to be built are collected from any available source. The overall requirements elicitation objectives for software engineering remain unchanged when applied to Web systems. However, the specific objectives for Web systems become: (1) the identification of content requirements, (2) the identification of the functional requirements in terms of navigation needs and businesses processes, and (3) the definition of interaction scenarios for different groups of Web users.

Requirements specification consists in producing a description of the requirements. Different techniques can be used for the specification: from informal textual description to formal specification techniques.

Finally, requirements validation consists in checking the requirements specification in order to establish whether the clients and Web application users needs are fulfillefd.

Escalona and Koch [3] are proposing a requirements engineering process with these three main activities for requirements: requirements elicitation, requirements specification and requirements validation. When corrections are applied to requirements, given there are some requirements not validated in the validation phase, the activity flow is returning to the specification phase refining and adjusting the requirements to meet the specifications, as presented in Figure 1. The phases of the requirements lifecycle are handled by various persons, such as: specialists (designers, analysts), groups (e.g. in JAD) or default actors (the project manager, the client).

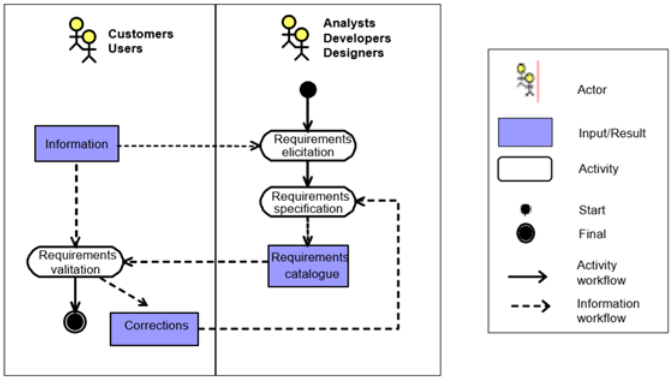

FIGURE 1. The requirements engineering process by Escalona and Koch

Another requirements lifecycle model proposed by Craig [4] specifies the requirements primary source, requirements owner, requirements location, the validation and their focus. The model covers requirements from project concept to testing and to deploy phase.

A study conducted by Heijstek and Chaudron [5] in 2008 over industrial practices of software development validates the level of effort assigned to each discipline during of the software development process as is described by the Rational Unified Process (RUP). We can conclude from the Figure 2 [6], which indicates the level of effort for requirements in RUP, that the requirements workflow is important in all the phases of a software project, mentioning here the Inception, Elaboration, Construction and Transition phases.

Notable in the RUP is that the analysis and design discipline is connected with the translation of requirements to a formal design. This type of design models can be considered tracks to be followed for writing the source code. A

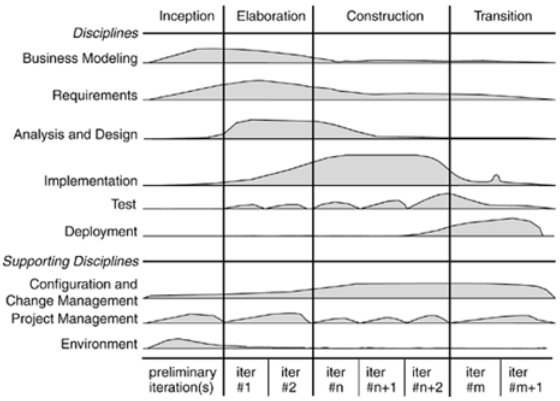

FIGURE 2. The Life Cycle for the Rational Unified Process

modeling language such as the Unied Modeling Language (UML) can be used to design classes and structure them into packages with well defined interfaces.

In order to connect the requirements to the Software Architecture (SA), Liu and Mei [7] are proposing a feature-orientation mapping from requirements to the SA. Their process implies requirements specification being organized as features. Several activities have to be performed on features: feature analysis and organization, feature elicitation, feature refinement. After that a mapping is perform from the feature model to the architecture models. The SA is defined from three viewpoints: as conceptual architecture, as logical architecture and as deployment architecture, each taking on certain types of features.

Another feature-architecture mapping (FArM) is presented by Sochos, Riebisch and Philippow[8]. Their method provides a mapping between features and architecture, which is based on a set of transformations on the initial product line feature.

The separation of concerns method is applied by Chen, Liu and Mencl [9] on Requirements Modelling. Even, though they split the model into several parts their approach is supporting separation of concerns and consistent and incremental modelling of requirements.

The separation of concerns is taken to a higher level by Moreira, Rashid and Araujo[10], in a multi-dimensional separation of concerns. Despite of the fact that they give up on using viewpoints, use cases or themes in representing the requirements, the solution provided is conceptualized in such a way that requirements are no longer scattered in different representations, but are using a unique representation. All requirements are decomposed in a uniform pattern regardless of their functional or non-functional nature.

Concern identification is based on the fact that certain concerns, both functional and non-functional, are repeating during system development. This kind of concerns may include shopping, booking, availability and security, so both functional and non-functional concerns.

The requirements space is divided into the system space and meta concern space. The system space gathers different types of systems that are possible to be realized (i.e. requirements associated to application that are just part of the requirements space); while the meta concern space comprises an abstract set of typical concerns, functional and non-functional, that are found in various systems (i.e. divides requirements into groups associated to an available concern in the space: authentication, navigation, mobility, portability,...).

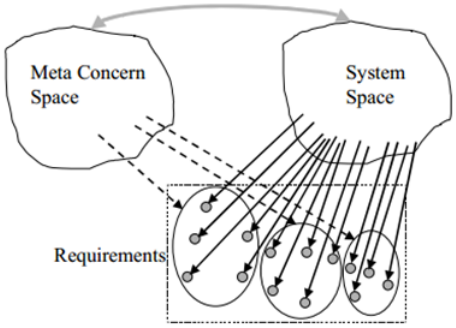

FIGURE 3. The system space and meta concern space

In the Figure 3, we can notice the requirements grouped in concerns. The concerns are part of the abstract meta concern space, while requirements are part of the concrete system space. By grouping the requirements in concerns, a conceptual binding is created between the two spaces. Both spaces are being represented using XML templates as in Figure 4.

```
<?xml version="1.0" ?>
<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 4. *InformationRetrieval* meta concern in XML

Requirements are recorded in concerns and are allocated unique ids. Also refined sub requirements are recorded the same way, but they are nested to the parent requirement, like in Figure 5. Also, a set composition rules is established to define relationships between concerns.

```
<?xml version="1.0" ?>
<Concern name="InformationRetrieval">
  - <Requirement id="1">
      It should be possible to retrieve information from the system.
      <Requirement id="1.1">It should be possible to access
          information about the attractions. </Requirement>
      <Requirement id="1.2">It should be possible to access
          information about the current location. </Requirement>
    </Requirement>
    <Requirement id="1.3">It should be possible to obtain a list of
        available preset tours. </Requirement>
</Concern>
```

FIGURE 5. *InformationRetrieval* concern in XML

A separation of the requirements in concerns is providing means in selecting the ideal or most suitable architectural choice for each concern, as presented in Figure 6. The architectural choices are usually not the same one, being possible to be even conflicting from one to another, but and analysis and negotiations with steakholders can lead to best accepted solution.

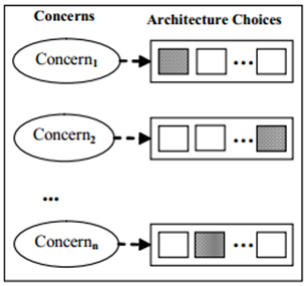

FIGURE 6. Architectural choices to satisfy each concern

3. Separation of Concerns in Web Enginnering

Mapping each requirement to the software architecture and eventually to the related developer is the last phase before the implementation of the requirements. The implementation of the requirements in the application phase should be added in the process. It is a natural step and is performed, indeed after the requirements are defined. So, in the process proposed by Escalona and Koch, the Requirement Implementation (Developing into the code) step is added, as it can be visualize in Figure 7. In this step is also included the process of the mapping of the requirement with the developer. The developer of each requirement is to be selected in conformity with the implementation method (database setup, html design, server-side script, etc.).

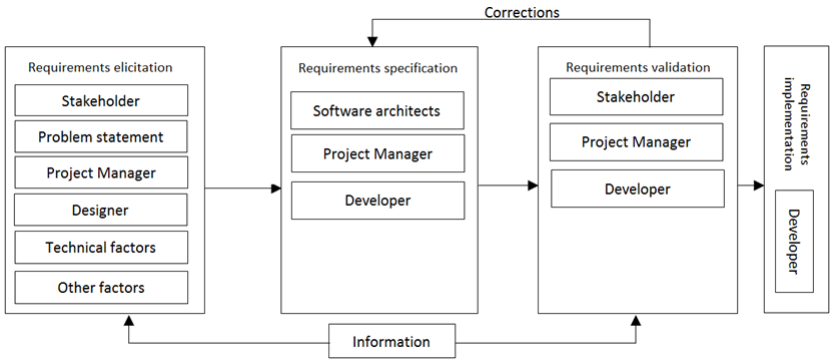

FIGURE 7. The updated requirements life-cycle and the actors involved

Developers are not involved just in the last phase; they are involved in all requirements processes. By developers we understand web designers, software architects, programmers, testers, tools of deployment or any other actor that directly affects the product. Developers are assigned specific roles in the product specification. Web designers can elicit requirements by sketching and storyboarding and they can specify requirements by modeling prototypes, while the programmers are being responsible to code prototypes in the requirements specification and in the requirements validation phase, as is presented below, in Figure 8.

Requirement elicitation	Requirements specification	Requirements validation	Requirements Implementation
Interviewing	Natural Language	Review	
JAD (Joint Application Development)	Glossary and Ontology	Audit	
Brainstorming	Templates	Traceability Matrix	Tool Modeling
Sketching and storyboarding	Prototypes	Prototyping for validation	Coding, Configuring, Designing
Use Case modeling	Use Case Modeling		
Questionnaire and Checklists	Scenarios		

Developer activities

FIGURE 8. Developer involvement in some of the main techniques used in the RE

Managing the requirements is a defining activity of the software development lifecycle, from the problem statement to the final product. Requirements have to be reflected in the product, therefore, implemented by the developers. After the requirements are set and ready, they are to be assign to be implemented. The process of implementing the requirements in the final product has to be assigned to a developer. The type of the developer is linked to the specific of the requirement.

Without a specific approach of mapping requirements to the software architecture the process of implementing the requirements is not fluent. Also, by means of implementing, there can be different developers, each one with their own expertise. If in enterprise applications are using a set with just several technologies, in web application there are considerably more technologies, and they are changing and updating constantly. Some technologies are new and in progress to be formalized, others will maybe be developed in the near future, providing solutions to the new communication and devices on the market.

The technologies used in web applications are included but not limited to: basic client side coding (html, javascript, css), advanced client side coding (Ajax, jquery, smarty), server side coding (java, ruby, .NET, php, asp, perl, python), client side and server side (tools to maintain complex javascript front-end applications), database technology (mysql, mssql, apache, sql lite, Microsoft SQL Server), web development software and frameworks (Macromedia Dreamweaver, WebDSL), content management systems CSM (wordpress, drupal, joomla), security (ecommerce, ebanking, networking), Web design tools (photoshop). Some developers master all technologies, but some are mastering just some of them.

A desirable approach into implementing the requirements is to assign the requirements to their appropriate technology, as represented in Figure 9. This will also help the project management in the development process to assign the requirements to developer teams in a methodological manner, so each requirement will be assigned to the best developer or team for implementation.

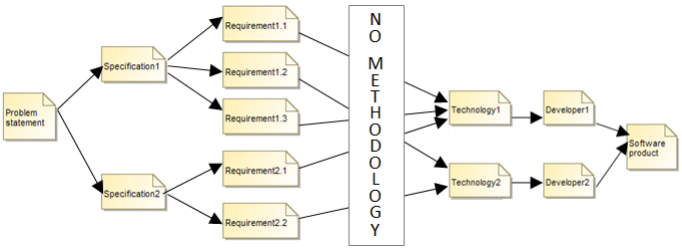

FIGURE 9. Information flow - direct connection from requirements to the technology used in implementation process

To meet this necessity, an approach can be elaborated, based on the multi-dimensional separation of concerns introduced by Moreira, Rashid and Araujo [10]. Their methodology can be transformed to sustain multiple aspects of the implementing the requirements process and not only. They do also provide tool support through their ARCADE tool.

One way is to create concerns spaces that are related to certain technologies. In order to do that, the requirements had to be assigned concerns into a

Meta Concern Space that can have concerns such technology types used in implementation. Following that join, a certain technology can be easly associated to a certain developer type, as shown in Figure 10.

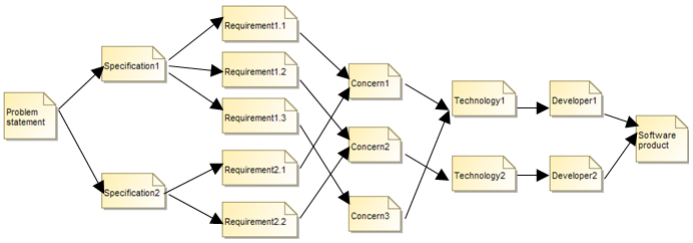

FIGURE 10. Assign requirements to technology using separation of concerns

To support this representation, to the meta concern in XML is added a *Technology* tag to record the technology appropriate to implement the concern (Figure 11).

```
<?xml version="1.0" ?>
- <MetaConcern name="InformationRetrieval">
        <Technology>Ajax calls</Technology>
        <Description>The operation of accessing information from a
                computer system </Description>
        <Examples>Database retrieval, Multimedia retrieval</Examples>
        <Relationships> Availability, Mobility, InformationUpdate </Relationships>
</MetaConcern>
```

FIGURE 11. The *InformationRetrieval* meta concern in XML with Technology details recorded

An upside of this method is that in the case of changing of one requirement, only one technology will be used and only one type of developer or one group will participate in updating the software product. Another plus of this approach is that in certain situations the technologies can have tools to perform the implementation of the requirements. In this way certain implementation does not have to be assigned to developers. Examples of tools for web are: CMS (Content Management System), WebDSL, Eclipse UML Class diagram. In other domains, other tools more appropriate can be used, like DOORS as a tool used in industry. In Figure 12 is presented such a situation.

Another possibility, depicted in Figure 13, is to create a concern space (MVC) that is assigned as concerns: Model, View and Controller, instead of technologies. This approach is useful when the level of conceptualization is high and this need is relevant to the project manager in the life cycle of developing the software product.

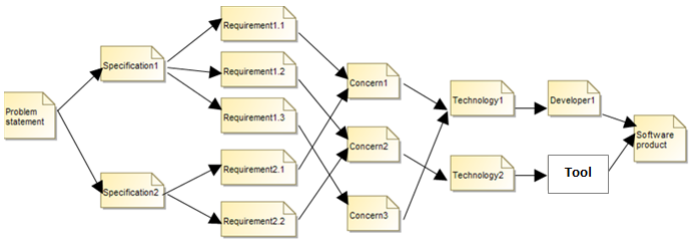

FIGURE 12. Concerns linked to development technologies relying on tools

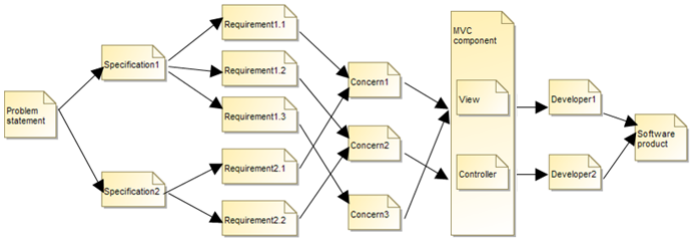

FIGURE 13. Assigning the requirements to MVC components in order to implement requirements

To support this representation, in Figure 14, we added to the meta concern in XML a *MVCComponent* tag to record in the MVC component appropriate to represent the concern

```
<?xml version="1.0" ?>
- <MetaConcern name="InformationRetrieval">
        < MVCComponent >controller</ MVCComponent >
        <Description>The operation of accessing information from a
                computer system </Description>
        <Examples>Database retrieval, Multimedia retrieval</Examples>
        <Relationships> Availability, Mobility, InformationUpdate </Relationships>
</MetaConcern>
```

FIGURE 14. The *InformationRetrieval* meta concern in XML with *MVCComponent* detail recorded

Different meta concern spaces can be created, such as MVCController meta concern space, with the concerns: model, view and controller. So, the requirements space can be altered by adding meta concern spaces. This creates a conceptual binding between the abstract representations of sets of concerns in different meta concern spaces. This binding is basically a mapping from one meta concern space to another, through requirements (Figure 15). These metaconcern spaces and the bidings between can be valuable for traceability purposes.

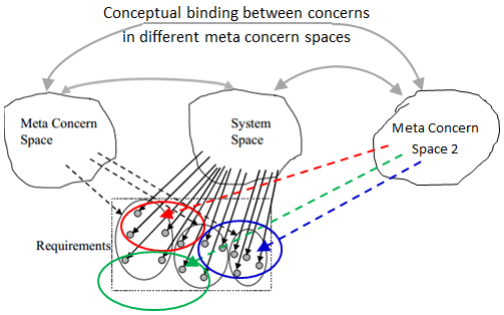

FIGURE 15. The system space and meta concern spaces

This model of dividing the requirement space can be extended to other object spaces (e.i. specification). There is a possibility for different system spaces to share the same Meta concern Space (i.e. specification space and requirements space can share the same Meta concern space, each of them being related to the same concern, such as *DevelopmentPriority*). That way a binding between different systems spaces is created, as shown in 16.

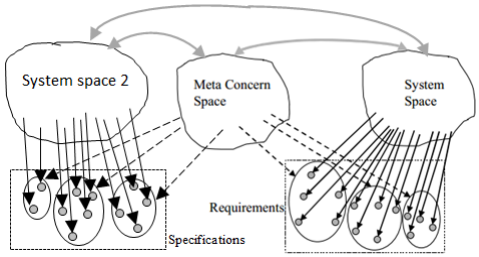

FIGURE 16. System spaces sharing a meta concern space

4. FUTURE WORK

Further investigation and analysis on the approach introduced in this paper could lead to valuable solutions or improvements to software engineering existent activities and methodologies, not only to the requirements engineering field or web application methodologies. One important step is to define rigorously the methodology and then to test it.

Tools can be developed to aid the process flow in using this methodology. Special tools could be developed to visualize the graphs created in the bindings between spaces. In some way, the bindings created in between different systems can be interrogated using a dedicated query language. The definition of sets of algebraic operations and operators on concern spaces can also be investigated.

5. Conclusions

This paper has proposed an approach based on the existing multi-dimensional separation of concerns in requirements engineering, with an impact in the implementation of the requirements for web applications. The proposed approach is subject to improvements, but it offers a solution in the process of selecting the technology of requirements implementing in the application and also can use the resouces and the tools provided by the original approach.

Other uses of the methodology are presented, such as the extension of the approach over multiple meta concern spaces and multiple system spaces. Also, the methodology introduces a different approach of the analysis of the requirements specification from a separation of concerns perspective and the traceability that is conferred to the requirements by this approach.

Acknowledgements: This work was possible with the financial support of the Sectoral Operational Programme for Human Resources Development 2007-2013, co-financed by the European Social Fund, under the project number POSDRU/107/1.5/S/76841 with the title Modern Doctoral Studies: Internationalization and Interdisciplinarity.

References

[1] Lowe, D., Hall, W., *Hypermedia and the Web. An Engineering approach.* John Wiley and Son, USA, 1999.
[2] Microsoft, *Microsoft Application Architecture Guide.* 2nd edition USA, 2009.
[3] Escalona and Koch, *Requirements Engineering for Web Applications A Comparative Study - Journal of Web Engineering.* Vol. 2, No.3, Rinton Press, USA, 2004.
[4] Chris Craig, *The requirements lifecycle.* http://businessanalyst.wikia.com/wiki/the requirements lifecycle USA, 2007.
[5] Werner Heijstek and Michel Chaudron, *Evaluating RUP Software Development Processes Through Visualization of Effort Distribution.* Leiden, Netherlands 2008.
[6] Aked, Mark, *RUP in brief.* IBM, 2003.
[7] Dongyun Liu, Hong Mei, *Mapping requirements to software architecture by feature-orientation.* Beijing China 2009.
[8] Periklis Sochos, Matthias Riebisch, Ilka Philippow, *The Feature-Architecture Mapping (FArM) Method for Feature-Oriented Development of Software Product Lines.* Germany, 2006.
[9] Xin Chen, Zhiming Liu, Vladimir Mencl, *Separation of Concerns and Consistent Integration in Requirements Modelling.* Macao, China, 2007.
[10] Ana Moreira, Awais Rashid, Joo Arajo, *Multi-Dimensional Separation of Concerns in Requirements Engineering.* IEEE, 2005.

(1) Babeş-Bolyai University, Department of Computer Science, Cluj-Napoca, Romania
E-mail address: calin.gal-chis@ubbcluj.ro