THE RATIONAL UNIFIED PROCESS WITH THE "4+1" VIEW MODEL OF SOFTWARE ARCHITECTURE - A WAY FOR MODELING WEB APPLICATIONS

Tarmo Robal, Vladimir Viies, Margus Kruus
Department of Computer Engineering at Tallinn Technical University

Abstract: The general purpose of any website is to attract visitors by its information. Information, which needs to be up to date, is only one of the components that make visitors to come back to the site again. Besides information presented, a site has to have a reliable architecture, irresistible user interface with clear and logical navigation through the information presented. This paper describes how to use the *Rational Unified Process* with the *4+1 View Model of Software Architecture* for modeling Web applications. The method described in this article, has been prosperously applied on the developing of the dynamic Web site of Department of Computer Engineering at Tallinn Technical University and therefore its practicability is proven.

Key words: Web site design and modeling, navigation control, data model, Rational Unified Process, the "4+1" View Model of Software Architecture.

1. INTRODUCTION

Over the past ten years a rapid development in the field of Web applications has occurred. In order to get the best out of a Web application, we need to somehow model it, before we can actually develop it in real life. Modeling Web applications is not just drawing nice pictures and adding some descriptions as a text. The process itself is very complex and needs certain methodology, not only entraining traditional software development processes, but also creative mind and short reaction time to changes. Web

applications should have visually attractive and irresistible user interfaces as well as reliable architecture. Typically there are three layers to provide the services: a data layer, an application layer and a client. It is a common knowledge that users are impatient; therefore, if the Web application is poorly designed or the information is not up to date, it takes only a minute or two to loose a customer because of frustration. Web applications have become a powerful media.

This article focuses on two methods: *Rational Unified Process* and *The 4+1 View Model of Software Architecture*. A way to model dynamic websites, using a combination of these two methods, is provided.

2. WHY RATIONAL UNIFIED PROCESS?

The Rational Unified Process is a software engineering process, which is an extension to *Unified Modeling Language* (UML) – a guide to the effective use of the UML for modeling. It organizes software projects in terms of workflows and phases, each consisting of one or more iterations, in the way that testing and validating design ideas as well as decreasing risks is possible in the early steps of a lifecycle. It is a framework that also can be implemented on iterative Web applications modeling.

The six reasons, why it is practical to use RUP for modeling Web applications, are as follows:

1. *Developing Iteratively* – it enables to take into account changing requirements, which in the case of Web applications are very variable, moreover, unstable, because immediate reaction to the changes in the trends, markets, customers' wishes is required. When developing iteratively, software elements are integrated progressively; therefore risks can be discovered and eliminated during integration, when it is easier and less expensive than facing faults in the final product. The functionality of a Web site can be added step-by-step; errors can be corrected over several iterations, which makes the architecture become more robust.
2. *Managing requirements* – it enables to have a better control over sophisticated Web projects, in order to improve quality and customer satisfaction. Capturing the requirements of a Web application on use cases, makes it easy to follow the real needs of customers through the whole development process of an application.
3. *Using component-based architectures* – it is evident that the architecture of Web applications has to be open and extensible. This can be attained by using modular architectures consisting of integrated components. Component-based architectures allow to

organize testing around single components, then expand it to larger set of integrated components and finally test the whole system itself.

4. *Modeling software visually* – the UML gives standard means of system description. It is important to have visual models of a Web application for controlling the changes and having a better overview of a system.
5. *Quality verification* – in most cases Web applications are for common usage, therefore failures can have a devastating effect – weak performance and low reliability means a loss of customers. Therefore, verifying the quality of a Web application and its components is vital.
6. *Control over changes* – Web applications consist of many components, which in layers can be described as *a client*, *an application layer* and *a data layer*. During iterative development many products are often modified, so it becomes vital to have a control over changes.

2.1 Phases of RUP for organizing software development

The RUP organizes the software lifecycle along five phases, seen from the management perspective, see Figure 1.

1. *Inception* – the aim of this stage is to attain a clear specification of the end- product vision, system requirements and main functionality described by use cases.
2. *Elaboration* – the main goal of this stage is in-depth analysis of the problem domain, defining system architecture and system use cases. As there are several types of Web pages and many different possibilities to develop Web, a clear vision from the *Inception* phase helps to coherently distinguish the technology to be used for the Web application development.
3. *Construction* – this stage is broken into several iterations, the product is developed iteratively until it is ready for distribution. The functionality of a Web Application is added step-by-step, mitigating risks and verifying quality.
4. *Transition* – at this stage the product is transferred to user's environment, necessary maintenance is provided.
5. *Evolution* – as Web solutions have to develop continuously and more rapidly, it is evident that the process is started again at the inception phase, for a new lifecycle.

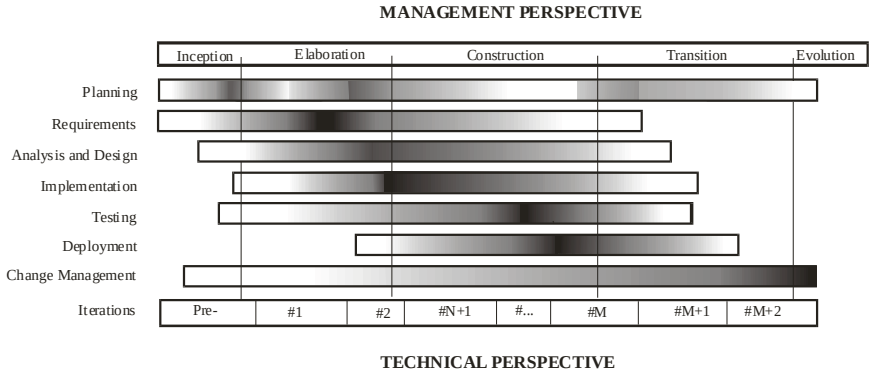

Figure 1. Software development organization in Rational Unified Process.

3. THE 4+1 VIEW MODEL OF SOFTWARE ARCHITECTURE

The 4+1 View Model of Software Architecture organizes a description of a software architecture using five concurrent views, each of which addresses a specific set of concerns. Developers capture their design decisions in four views and use the fifth view to illustrate and validate them. This use of multiple views allows to address separately the concerns of the various "stakeholders" of the architecture: end-users, developers, system engineers, project managers etc. and to handle separately the functional and non-functional requirements.

The 4+1 model consists of the following views, see Figure 2:

- *Logical view* – presents a system from the point of view of end-user,
- *Development view* – description of a system for programmers and managers,
- *Process view* – representation of functionality, performance, scalability and the possibilities for integrating with other systems,
- *Physical view* - the product is presented from the point of view of end-user.

The description of software architecture is described using these 4 views and illustrated by scenarios, the fifth view, which interweaves it all together.

For each of the views the Duane Perry and Alexander Wolf's software architecture formula is applied on independently.

Software architecture = {elements, forms, principles/constraints}

Hereby, each view has its own elements, forms, patterns, principles and constraints applied on it. In the following chapters a way for modeling Web applications using *RUP* with the *"4+1" View Model of Software Architecture* is being discussed accompanied with some illustrative

examples from the design of the dynamic Web page of Department of Computer Engineering at Tallinn Technical University.

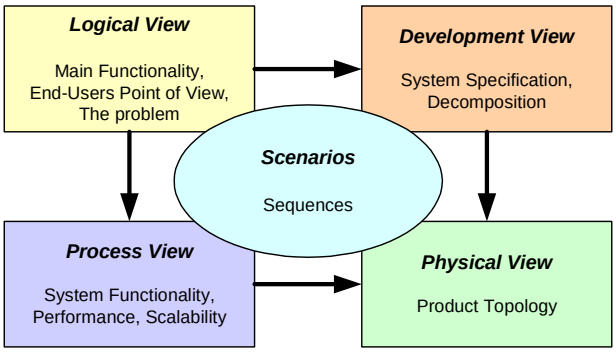

Figure 2. Software architecture description with 4+1 views.

3.1 Web design with RUP in the "4+1" View Model

Efficient Web site development needs quick and careful planning, in order to model a site that has a reliable architecture in conjunction with flexibility and extendibility. In other words, there has to be a clear and overwhelming vision, so that the problem domain, the system, its main functionality and features could be defined. For Web applications modeling the UML is recommended to use.

3.1.1 The Logical View

As soon as the vision of a future Web application has been reached, the functionality of the system can be described using classes and use-cases.

The main goal of the logical view is to:

- Clearly define the task to be solved,
- In-depth analysis of the problem area, in this case the Web site,
- Coherently represent the functionality and the users of the system, using use-case diagrams, as shown on Figure 3,
- Define the objects and their possible states in the system as classes and states,
- Develop the user interface *Creative Design Brief* and navigation map, see the User Interface Development chapter.

Non-functional requirements and a glossary of terms to be used are also composed in the logical view.

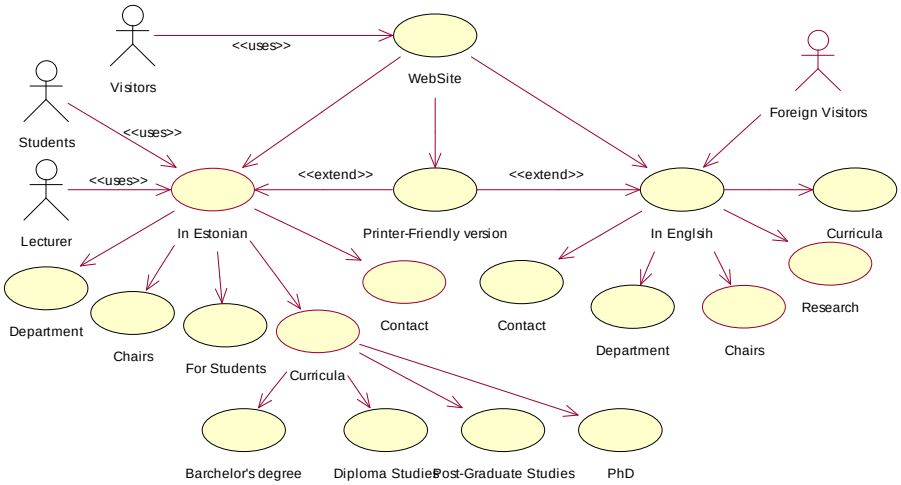

Figure 3. System functionality. Users.

3.1.2 The Development View

As the requirements and functionality are already described in the logical view, the organization of actual software modules is represented by module and subsystem diagrams as shown on Figure 4. The aim of the development view is to specify:

- The media for the realization – a set of applications that are required to construct a Web application; i.e. Apache server with PHP4 module,
- Software re-use issues,
- Constraints to the system further development,
- Logical data model,
- The User interface *Creative Design Comps* and *Web Design Elements*.

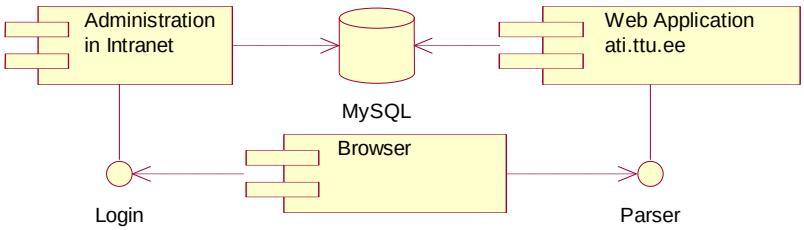

Figure 4. System components.

3.1.3 The Process View

Concurrently with the development view, issues such as concurrency and distribution, integrity of the system and fault-tolerance are elaborated. The scope of the view is to describe:

- Functionality to be realized – classes and methods that actually realize the specified functionality, see Figure 5,
- Non-functional requirements concerned with the implementation of the functionality,
- User interface initial prototype, full prototype and full navigation map.

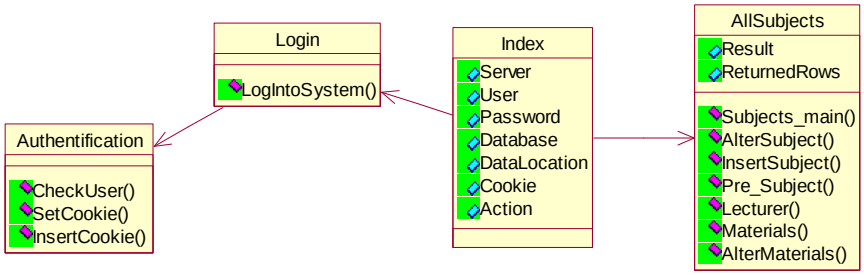

Figure 5. Functional specification (fragment).

3.1.4 The Physical View

The physical view brings together all the elements from logical, process and development view, taking into account the system's non-functional requirements such as reliability, performance, scalability; several different configurations might be used for the Web application, some for testing, others for system deployment for various customers. The actual realization of data model for a specific database system is also submitted.

3.1.5 Scenarios

Scenarios are *putting it all together*. Interaction between objects in the system is expressed by object interaction and scenario diagrams as shown on Figure 6.

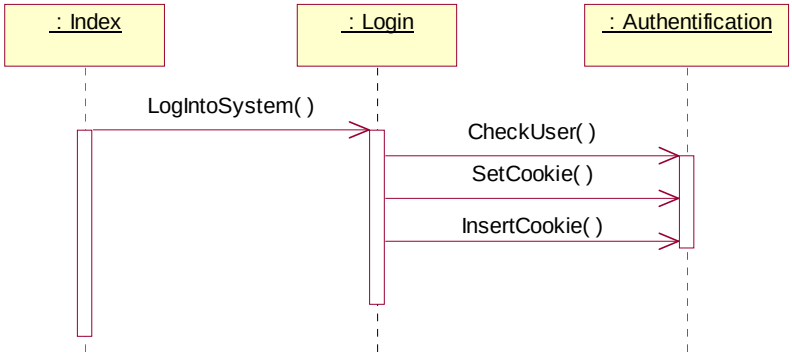

Figure 6. Scenario. Logging into system.

4. USER INTERFACE (*UI*) AND NAVIGATION CONTROL

One of the most important things in website design is the graphical user interface and the navigation organization through the site. As for the users, they can be manipulated by the look of interface and vice versa, the way the UI is organized, affects on how users react. Therefore, the first impression is very important, otherwise there is a high risk of loosing customers. A good navigation organization ensures that users can easily and quickly find the information they need. Moreover, since most sites have a hierarchical structure, the navigation helps users to navigate in the hierarchy, which also produces knowledge and information. In addition to site's main navigation, indices, search engines and sitemaps are also the source to reach the required information. The aim of the navigation is to provide users with feedback on what can be found on the site and how it is organized.

4.1 The User Interface Development

The UI development is an iterative process, which can be divided into seven main steps, as follows:

1. *Creative Design Brief*, which defines the mood of the site (provocative, playful site, etc), browsers to be used, site's structure (frames, formatting, etc.), requirements to graphics and colors.
2. *Navigation Map* – a view of the organization of navigation for a site. The initial navigation map evolves from the use-case model discussed under the logical view. Each level of the schema presents the number of clicks users have to perform before they reach their target.

3. *Creative Design Comps*, which are mock-up pictures of what a site might look like, provide stakeholders with site's visual solutions. The selection of an appropriate solution is an creative and iterative process.
4. *Web Design Elements* specify unique and standardized graphical elements *i.e.* images, buttons, lines, logos, in order to guarantee coherent style of the site.
5. *Initial User Interface Prototype* is developed according to the selected design mock-up. The main functionality is realized according to the features described in the logical view.
6. *Full User Interface Prototype* – initial prototype is developed so that it covers all of the functionality.
7. *Full Navigation Map* – as soon as the full interface prototype is introduced, a complete navigation map for the site is composed, describing all known pages in the site, see Figure 7.

Steps 1, 2 and 7 represent logical structures; steps 3-6 address the issues of actual layout of a Web application.

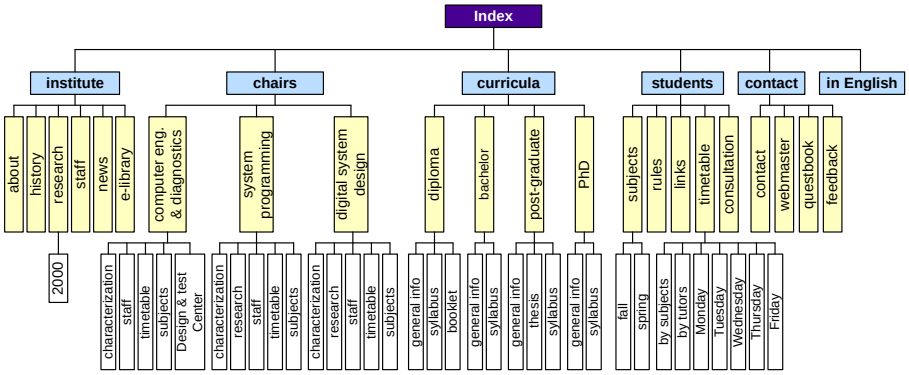

Figure 7. Full navigation map.

4.2 Forms of Navigations

Once the *Creative Design Brief* is reached, the organization of site's navigation is considered. There are many different forms of navigation, four of which we consider here are *the top*, *the vertical*, *the top-vertical* and *the split navigation.*

The Top Navigation is one of the simplest navigation schemes, where menu items are placed on a horizontal bar as shown on Figure 8. Top navigation can be described by the following characteristics:

- Menu present on every page; selected menu items highlighted,

- Vertical place on the page is not consumed by menus, therefore the content can be displayed across the whole page,
- Possibility to display related topics in the margin column,
- The maximum number of items is limited to 6-8, because of the width of a page,
- The maximum number of menu levels is limited to 2-3. As each level is placed underneath the previous one, the content area is pushed down.

The Vertical Navigation is another very widely used navigation form. The page is split into at least two columns; one of the columns is used for the navigation, see Figure 8. Generally the menu is organized in the left column, although, depending upon the vision of the site, other solutions are also possible. Since menu items are placed underneath each other, having a large number of items may lead to more scrolling. In a case like that, the solution might be to exploit the top-vertical navigation.

Neither top nor vertical navigation meet the real requirements, in particular when there are more than two menu levels and several item groupings. **A Top-Vertical Navigation** is a combination from vertical and top navigation, where menu levels can be organized either in a vertical column or on a horizontal bar, as shown on Figure 8.

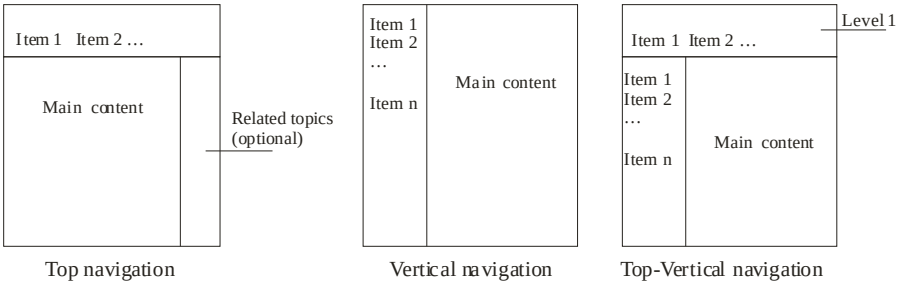

Figure 8. Main navigation schemes.

In case the top-navigation cannot fulfill the need for navigation organization, **Split Navigation** should be considered. With the split navigation the first level (or couple of first levels) is separated from the remaining ones. In other words, defining the first level as level *n*, the second as *n+1* etc, after a selection on level *n*, the first level is level *n+1*.

4.3 Navigation Suitability and Efficiency Analysis

A research on the navigation organization's efficiency and suitability was carried out on the website of Department of Computer Engineering at Tallinn Technical University during 01-12 February 2002, the beginning of a new semester. The dynamic website under analysis was modeled using the

methodology described in this article. The aim of the study was to verify the methodology through its quality control employing the results of the site's usage.

The website is dynamic and is organized using *Top-vertical navigation* with three different navigation levels. During the aforementioned period, the requested pages' ID, request time and the IP number of a referring computer were logged for analysis. In the log 5560 requests were protocolled, 1588 of which remained suitable for the analysis after refining. The necessity of refining the results was caused by the web site usage by students in the computer classes of the department. The analysis performed were:

- Search combination analysis in sessions,
- Characterization of searches in one session,
- Average time between two clicks according to the menu level.

The ***Search combination analysis in sessions*** divided the operations performed on a web site into five groups according to the number of clicks performed during session. The session is defined as a sequence of request made by one user in one timeslot (visit of a site). The five groups were:

1. *Group one* – information requests with one operation; users have performed only one click, either on level 1 or level 2 to reach their target.
2. *Group two* - information requests with two operations; users have performed two clicks, either on level 1, level 2 or level 3 to reach their target.
3. *Group three* - information requests with three operations; users have performed three clicks, either on level 1, level 2 or level 3 to reach their target.
4. *Group four* - information requests with four operations; users have performed four clicks, either on level 1, level 2 or level 3 to reach their target.
5. *Group five* - information requests with more than four operations; users have performed more than four clicks, either on level 1, level 2 or level 3 to reach their target.

The results of the analysis are shown on Figure 9.

The analysis of ***Characterization of searches in one session*** showed that the majority of users made only one operation on the site during one session, see Figure 9. In 21% of cases users performed more than one requests per session. This fact can be justified by multiple information needs or by unsatisfactory results of a request. In 4/5 of cases the necessary information was reachable immediately.

The timing analysis of sessions and operations had the statistical results described in Table 1. The research results indicate that users reached their requested target information approximately in 22 seconds. The minimal search time of 20 seconds is with great probability reached by a user who is looking something not very specific and the maximum search time by a user who has briefly explored all the levels preceding the required level 3 information.

Table 1. Timing analysis.

Average operation time:	22 seconds
Maximum search time (3rd level):	120 seconds
Minimum search time (3rd level):	20 seconds
Minimum time between two operations:	5 seconds

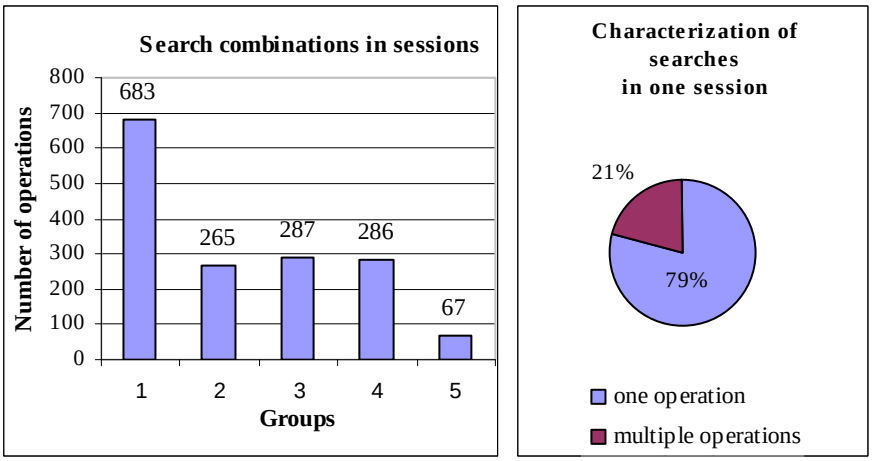

Figure 9. Searches and their combinations in sessions.

Figure 10 depicts the relation between operation time and the menu depth reached. Note that level 4 is actually not a level, but describes operations in the third level, the level, where only specific information is presented and requests for data are fast and simple. The time consumption on the first and the second level is caused because the information presented on these levels is mainly generalization of specific data presented on the third level. The request time of 28,3 seconds on the third level is caused by the navigation through the more general levels 1 and 2.The average time between two operations is calculated as $\Delta t = t_{click\ i} - t_{click\ i-1}$.

The analyses indicate that due to the logical and clear organization of the site the requested information is reached almost instantly. Therefore there is no necessity to rearrange the current navigation schema to minimize the number of operations.

The authors share the opinion that these kinds of analyses have perspective and continue data capturing for long-term analyses, trend and legitimacy studies.

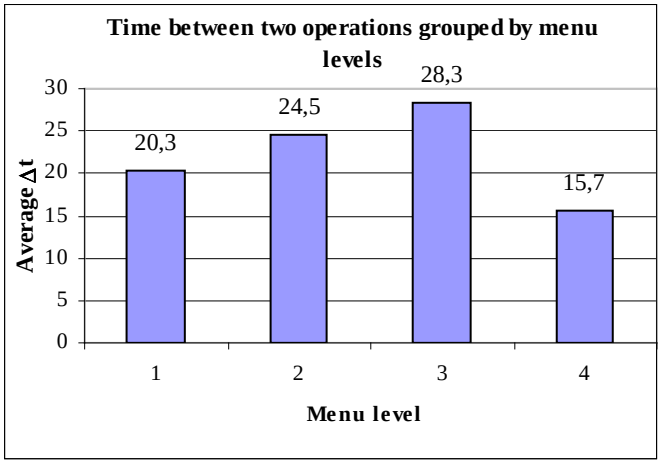

Figure 10. Average time between operations with reference to the depth of menu.

5. INFORMATION SYSTEM AND DATA MODELS

So far the methods which provide an easy and logical way to model websites have been discussed with user interface design and navigation organization, but the "heart" of a dynamic website, the information system, has been ignored. For flexible website administration, the information should be stored in a database. There are two main possibilities:

- To store all of the information in a database – very convenient way for data updating, but excessive programming work is needed to design the output on a website.
- To have only pointers that refer to the information stored in a database – a useful technique, if the information does not need updating. Moreover, it is possible to store pointers, which refer to classes and functions that are called to produce the information in reaction to user selections.

A recommendation would be to use the first method in conjunction with the second one.

The logical data model is iteratively developed from the conceptions of objects and data needs presented in the logical view. At this stage, the data

model should be normalized, the performance, indexes and foreign keys will not be considered, as they refer to the physical data model.

The physical data model, which represents the actual mapping to the database system, is elaborated from the logical model on the assumption of selected database management system. The physical data model can vary according to the chosen database system. At this phase all the necessary keys and data types are defined, also indexes, constraints, views and triggers. The model is optimized for performance, some de-normalization may occur in order to increase the system's performance. Developing physical data model is an iterative and experimental process. As a result the final model is created.

SUMMARY

The importance of Web applications in our lives has rapidly increased over the past ten years. The Web is an endless source of information; therefore there is a continuous competition to have the best possible web solution. Such solutions cannot be developed overnight, they need to have a clear vision, in order to begin their modeling, which is followed by the actual construction of a site. There are a lot of different methods on how to develop Web applications, one of which is described in present article, involves the *Rational Unified Process* with the *"4+1" View Model of Software Architecture*.

REFERENCES

1. Royce, W. Software Project Management: A Unified Framework. Addison-Wesley Professional, 1998, 448 pp.
2. Kruchten, P. The 4+1 View Model of architecture // IEEE Software (1995) vol. 12, iss. 6 pp. 42 -50
3. Kruchten, P. A Rational Development Process // Crosstalk (1996) vol. 9, no. 7
4. Kruchten, P. What Is the Rational Unified Process? Rational Software, 2001, 11 p.
5. Conallen, J. Modelling Web Application Architectures with UML // Communications of the ACM (1999) vol. 42, no. 10, pp. 63-70.
6. Ward, S., Koll, P. Building Web Solutions with the Rational Unified Process: Unifying the Creative Design Process and Software Engineering Process. A Rational Software & Context Integration White Paper (1999), 12 p.