Three view model of e-portfolio assessment system

Milen Petrov, Adelina Aleksieva-Petrova
Sofia University, Sofia, Bulgaria
email: milenp@fmi.uni-sofia.bg, adelina@fmi.uni-sofia.bg

Abstract

In this paper, we describe the software architecture for system which supports non traditional form of assessment named e-portfolio. The architecture model is described through three views: use case view, logical view and implementation view. Based on these three views software system was designed and developed. Some implementation issues are discussed.

Keywords

Use case view, logical view, implementation view, assessment model, e-portfolio, eclipse RCP, java

INTRODUCTION

Some new forms of assessment, various from existing traditional forms, as examinations and tests emerge with the development in the area of human abilities evaluation. These new forms have showed that they are only helpful to estimate the competences of the given human, because they affect aspects, which more sharply as well fully encompass the human competence. For example assessment of e-portfolio, 360 degrees feedback, peers assessment and so on.

The present work shows one of the new non-traditional forms of assessment (e-portfolio) which can be gained with software instrumentalities as well to be integrated in the entire process of assessment. The integration must cross as well with the existing standards and specifications as IMS [1].

There are many definition of e-portfolio and one of them is that an electronic portfolio is collection of electronic evidence assembled and managed by user, usually on the Web. E-portfolios are both demonstrations of the user's abilities and platforms for self-expression. [2]

According to IMS definition the assessment ePortfolios are used to demonstrate achievement to some authority by relating evidence within the ePortfolio to performance standards defined by that authority [3].

Therefore the main goal of the paper is to describe software architecture of the system which support assessment e-portfolio and have to demonstrate user's abilities, interests and achievements in a certain area through relating evidence.

The additional focus of current paper is software architecture description of model by following different viewpoints:

- **Use case view**: shows the main process as use cases. These use cases typifies the primary use cases that will be implemented and thus shows the most important process of the system.
- **Logical view**: gives the high-level system view from a user's viewpoint, by providing descriptions, pictures and diagrams of the functionality.
- **Implementation view**: shows the system's main components from a developer's point of view, by providing descriptions of frameworks and descriptions of significant classes.

USE CASES VIEW OF THE SYSTEM

In fig. 1 is showed all activities of the user and what he or she has to do when makes assessment of portfolio with designed software tools.

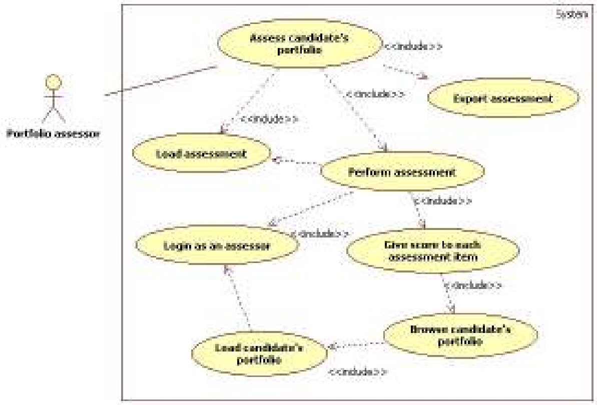

Figure 1. Use cases of the process of e-portfolio.

The main user is portfolio assessor who has access to the candidate's portfolio. According the selected portfolio the system support performed assessment or export the portfolio.

The use case "Perform assessment" includes the process of given score to each assessment item which is corresponded to the user's abilities and interests.

LOGICAL VIEW OF THE SYSTEM

The logical view in the architecture of the model shows system from stand of its users. It holds figures that illustrate function of the system.

The *Perform Assessment* Use Case and *Browse Portfolio* Use Case are used to create an analysis class diagram, and to show the concepts using boundary classes (the user interface), controller classes (business logic) and entity classes (data storage). The resulting diagram can be seen as a translation from user concepts to system concepts.

The analysis class diagram for representation on business-logic, made is on Fig. 2, showed of the base of the main use cases for the system, as well that demonstrates the basic concepts. This diagram gives a conceptual overview of the system's parts and is consequently helpful to both developers and software architects. The analysis classes give conception view on the parts of the system and help for understanding its architecture.

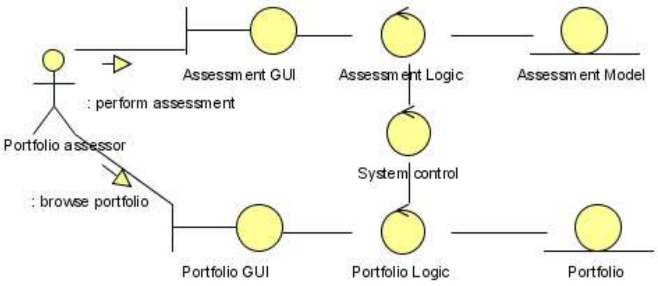

Figure 2. Analysis class diagram of the functionality of the system.

The main activity of the user of the system is *assessment of portfolio*. It is developed with *Assessment GUI* and *Portfolio GUI*. Under those two directions there is control logic (Assessment Logic and Portfolio Logic) and logic to manage of system and activities (System Control). On the other hand main classes (Assessment Model and Portfolio) corresponding for persistency data.

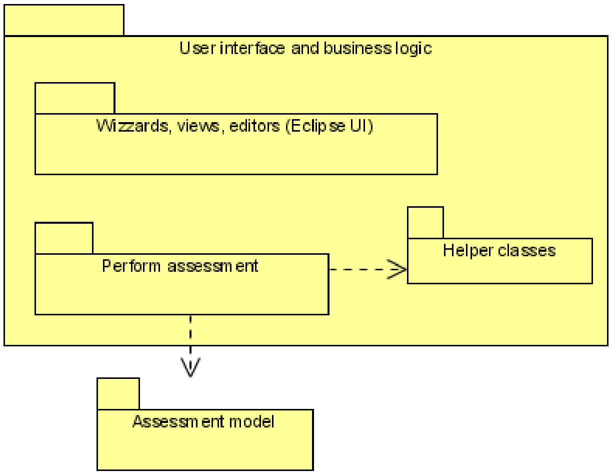
User interface and business logic
Wizzards, views, editors (Eclipse UI)
Perform assessment
Helper classes
Assessment model

Figure 3. Package diagram of the system's functionality

On fig.3 is showed a package diagram of the system. The main aim of that diagram is to describe all functionality of the application according the logical group. Package diagrams organize elements of a system into related groups to minimize dependencies between packages. The main packages are: assessment model, package represented GUI and business logic of the system.

The other package holds Eclipse environment for development of the graphic interface (wizards, views and editors) and API for SWT and JFace.

IMPLEMENTATION VIEW OF THE SYSTEM

The application is disposed in the form of several packages [4]. In each of them there are classes with defined functionality.

On figure 4 is presented package structure of the main package.

These packages contain implementation of model evaluation in the form of Java classes and the names and structure of these classes are as defined in the XML schedule file copies of the model.

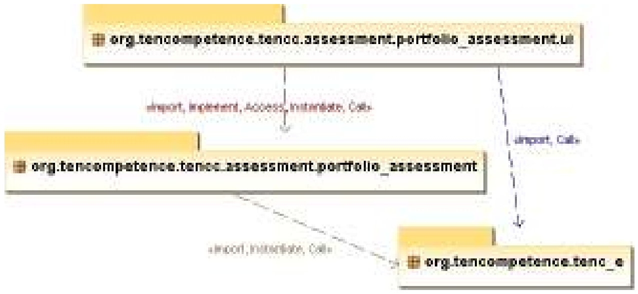
org.tencompetence.tencc.assessment.portfolio_assessment.ui
«import, implement, Access, Instantiate, Call»
«import, Call»
org.tencompetence.tencc.assessment.portfolio_assessment
«import, Instantiate, Call»
org.tencompetence.tenc_e

Figure 4. Package structure of the application

There are both classes that are part of the business logic of the application, and purely technical classes environment (framework) of the Eclipse RCP. Such a purely technical classes are:

- Activator (extends AbstractUIPlugin) – Eclipse plug-in has such class in the package which connect project with the environment of the Eclipse. It sets the context of the application as plug-in in the Eclipse environment, and provides methods hook, which may redefine the application to have access to specific events and resources of this environment. Typical methods are those who call upon loading and unloading of software in plug-in from the Eclipse environment.
- Application – the java class which controls the life cycle of the application. There is similar function of this of Activator but it only guarantee access to different type resources, such in terms of the overall project as feasible implementation.
- The following three classes: ApplicationActionBarAdvisor, ApplicationWorkbenchAdvisor and ApplicationWorkbenchWindowAdvisor, are used for access to the resources and event of workbench in Eclipse. These classes are conducted to manage different parts of this environment, like events, contextual menus, toolbars and others.

Classes, part of the pattern of application, are grouped in three different directions:

- Comprehensive data models of the system – it is implemented through class Model, which represents the model part of the logical organization of the system using the template design Model-View-Controller.
- Model Portfolio - implemented through classes *PortfolioSection* and *PortfolioSubsection*. They set the structure of the portfolio as arranged set of files. The class *PortfolioDAO*, which represents the realization of Java template design DAO, is used to extract portfolio database.
- Model of state of assessment - the different conditions in which the application passes in run-time by the user. It represents the realization of design templates State and Observer. It includes classes PerformAssessmentSession, ISessionListener and enumeration type SessionState.

CONCLUSIONS, LIMITATIONS AND BENEFITS

On base of these three views the system is designed and implemented. In fig. 5 is shown the screenshot of the system.

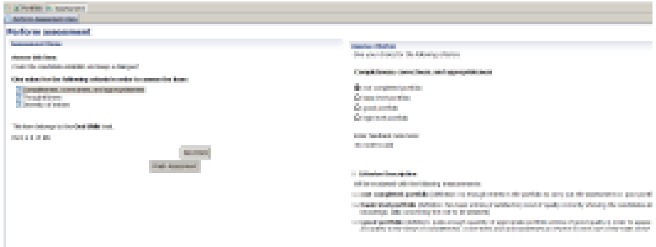

Figure 5: A screenshot from the proof-of-concept assessment player tool

The system has capability to import and parse xml file which is constructed according assessment specification. It offers to user to perform assessment activities and assigned to user role. After assessment activities is performed – it offers to the user to store results also in format, which is compliant to the assessment specification. This tool also can load already performed assessment activities, and to preview or to evaluate results from the activities, performed in previous assessment run.

The evaluation of the system is carried out with the black box testing on the basis of preparation of specific examples of evaluation (a copy of a XML format) and gamble in the system. There are numerous unit tests itself to copy the model for evaluation.

Estimates for the system were satisfactory. There is much that could be developed, but overall it provides the necessary functionality to carry out the evaluation, which is based on the model of TENCompetence Assessment Model.

In the process of developing the system it were made following key activities:

- Research and analysis of problem areas
- Defining the requirements of the system - certain portion of outstanding service and functional requirements of the system.
- Create a copy of the model evaluation - Detailed analysis of the model and clarification of its components and concepts enshrined in it. It was development XML scheme which presents copies of the model. The items in this scheme satisfy the concept of model evaluation, together with elements for modeling activities, roles and additional input for the evaluation. It was created a module for developing and storing copies of the model. It includes the scheme itself, the source code generated by the scheme through JAXB technology, unit tests of the instance.
- Choice of technology for development – It was selected Eclipse RCP, as a target platform for development and JAXB technology for automatic code generation. MySQL was choosing as a supporting database.
- Design, development and testing of the system – It was identified and developed individual components of the system topic on the architecture of Eclipse plugins for software and the necessary testing to ensure fairness of the system.

As conclusion we can say that developed tool(s) is based only on one type of non-traditional assessment, and conclusions therefore can be limited, nevertheless current research is very important step of assessment model and specification validation activities. Future work will include more widely adoption of other different methods of non-traditional assessment approaches.

ACKNOWLEDGMENTS

The work reported in this paper is partially supported by the SISTER project, funded by the European Commission in FP7-SP4 Capacities via agreement no.: 205030.

REFERENCES

[1] IMS Global Learning Consortium, *IMS Question & Test Interoperability Specification*, 2006

[2] Hebert, Elizabeth A., *The Power of Portfolios - What children can teach us about Learning and Assessment*, San Francisco: Jossey-Bass, 2001

[3] IMS Global Learning Consortium, *IMS ePortfolio Specification*

[4] Borovska P., M. Lazarova, *Methodology for Parallel Application Development*, Vth International Conference “Challenges in Higher Education and Research in the 21st Century”, Sozopol, Bulgaria, 5-9 June, 2007