Object-oriented Computer-aided Learning Design System Development

E. Shoikova, M. Ivanova

Key Words: *Computer-aided learning design System; learning technology standards; object-oriented design, unified modeling language; use case analysis.*

Abstract. *This paper describes the object-oriented development of a Computer-aided Learning Design System as a result of the project carried out at the Technical University of Sofia, the R&D Laboratory „eLearning Technologies". The system assists educators and other learning design professionals in their units of learning analysis and design activities and deliver effective support, promoting the exchange and interoperability of eLearning scenarios. The development methodology includes the Rational Unified Process as a successful development strategy; Unified Modeling Language for clarifying the requirements, architectures and designs, Use Case Analysis to elicit, organize, and document required system functionality; the concepts of forward software engineering to build the high-quality system. The system is realized as an integrated part of the open source standard-based eLearning platform ATutor.*

Introduction

Providing the right knowledge and tools within the learning design process for educators and trainers in our today's information/knowledge society to educate students in an effective and efficient manner have been heavily emphasized and recognized within higher education. Although there exist several high level authoring tools conformant to the IMS Learning Design specification, that facilitate the learning design process, these tools are not fully interoperable, it is too hard to make units of learning with them as well as a lack of a library of complete, well-founded, field tested examples has been identified.

Why is learning design important for professional educatiors? Learning design forces to the author-teacher think about pedagogy, guides authors to reflect on learning activities, tasks, assessments, interactions, skills, and tools for better achievement of learning objectives. Learning design supports authors to model effective educational process, personalized and flexible learning as a respond to the challenges of the 21st century skills: using information, managing resources, collaborating, problem-solving, communicating, and lifelong learning.

This paper examines the object-oriented development process of a Computer-aided Learning Design System (CALDS). The following tasks have been accomplished:

- State-of-the-art in learning design editors. On the basis of the comparative analysis of the current open source learning design tools done, problem definition and CALDS requirements have been set.
- Functional system specification was defined through Use Case analysis and learning design domain model was built as a visual representation of conceptual classes.
- The CALDS dynamic model has been created to present the interactions among objects which is important for object's interface definition.
- An Object oriented server-based CALDS architecture has been developed.
- The CALDS implementation model in an object-oriented PHP programming language, HTML, CSS, JavaScript has been made including a detailed design and construction of each CALDS component, then integration of the code and user interface in the eLearning platform ATutor.
- The testing has been performed in the following steps: Unit testing tests, Integration testing, Functional testing, System testing, Acceptance testing.
- Software prototyping is applied to create a model of the full-featured CALDS.

State-of-the-art in Learning Design Editors

Open source software makes important inroads into the world of learning, education and training. There is emerging support from open source SCORM virtual learning environments as Moodle 1.5, ATutor 1.5, dotLearn, Claroline 1.4, Fle3, Sakai 2.0, Colloquia 1.3.2.[1]-[4]. Many current projects are running with focus on open source Learning Design tools development. The 1st generation of IMS Learning Design authoring tools as RELOAD, CopperAuthor, Komposer, and aLFanet LD Editor, provides form-based interfaces for the definition of educational scenarios and/or units of learning by using the XML structure of the specification as the main driver of the authoring process. Their main advantage is providing direct control of the information model elements while their disadvantages are as follows: difficult to be used by less experienced designers and require pre-processing (outside the tool) of the structure of the desired scenario in order for a designer to be able to express it directly in XML notation. This first generation of IMS Learning Design-aware tools supports only part of the design process and they work in an isolated mode. Some tools explicitly address technically savvy users or present too complex pedagogies to expect easy to use authoring tools. The 2nd Generation of IMS LD Authoring Tools as MOT and LAMS, provides graphical-based, drag-and-drop interfaces for the definition of educational scenarios and/or units of learning. The main advantage is supporting the design process without requiring pre-existing knowledge of the details of the IMS Learning Design information model, while their disadvantages are the following: they generate the IMS LD manifest from a graphical representation of the learning flow but not the other way around; these tools are not capable of carrying out the transformation of the IMS manifest to the corresponding

graphical representation. These are used by educators interested in working with the tools to learn more about the IMS LD specification rather than to use them in educational production processes [5]-[8].

What do teachers-learning designers would get with IMS Learning Design-aware system automating learning design process and giving effective timely support?

- Every other advantage that a standard notation brings: reflection, communication, sharing, reuse, research, similarity studies, evaluation, etc.
- Integrate the large number of isolated existing standards (LOM, CP, QTI, RCD, LIP, ..) to create executable and interoperable units of learning ('courses').
- Support automation of the workflow in the teaching/learning process to decrease workload (especially of teachers).
- Should bring new more effective, efficient & attractive learning models (active learning, problem based, ...).
- Basis for the next generation of e-learning systems: increasing the 'richness' of different learning activities.

In order to respond to the identified needs, the proposed solution was developing the IMS Learning Design-aware computer aided system that assist educators and other learning design professionals in their units of learning analysis and design activities and delivering effective support, promoting the exchange and interoperability of eLearning scenarios.

Development Methodology

The latest software design and development methodologies are fundamentally changed in order to minimize the risk of failure and to deliver successful solutions faster [9]-[11]. The Rational Unified Process (RUP) with its four phases: Inception, Elaboration, Construction, and Transition is used as the main development strategy, because it relies on the iterative and incremental process model for implementing object-oriented analysis, designing, and programming concepts and constructing a complex software system that is realized as agile subsystems. The functional requirements, logical structure and behavior of the system are visualized through rigorous, traceable and maintainable models by using Unified Modeling Language (UML) [12].

The Computer-Aided Software Engineering Tool Enterprise Architect is used for automating the processes of forward engineering. It covers all aspects of the development cycle, providing full traceability from the initial design phase through to the deployment and maintenance. It also provides support for the testing, maintenance and change control.

Use Case Analysis

The Use Case analysis is used in the early stages of the object-oriented development project as the most efficient method for the CALDS requirements capturing [13]. The system functionality is presented from a user's perspective creating UML Use Case Model (*figure 1*). Also, the Use Case analysis helps to lay out the actors or users and their role in running the system. The main actors that interact with the CALDS are as follows: Engineer/Learning Designer, Engineer/Educator, Administrator and Learner - human actors and non-human actors — eLearning system ATutor, UML Tool and verification tool RELOAD. During interaction an actor generates events: *Create Unit of Learning, Get Support, Verify Unit of Learning, Start Unit of Learning, Store and Install* to a system, usually requesting some operation in response.

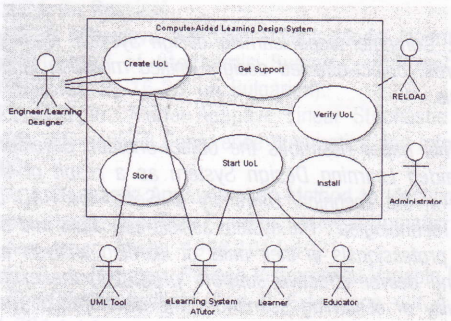

Figure 1. CALDS Use Case Diagram

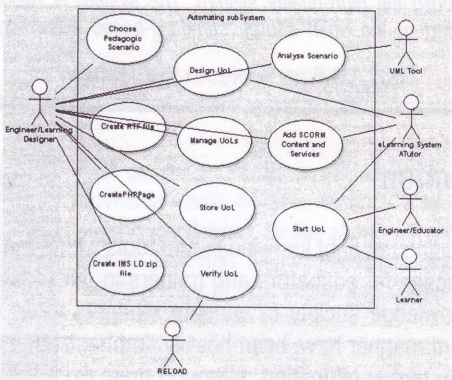

Figure 2. Use Case Diagram of the Automating subSystem

The CALDS is a large and complex environment that pursues two main goals: to automate the learning design process and to perform effective support to the Learning Designer. For this reason it is constructed from two coherent subSystems: Automating and Supporting.

In the *Automating subSystem* the learning designer chooses from a library a pedagogic scenario template presented as incomplete structures, designed to simplify the construction of activity sequences; designs the unit of learning using editor to create new learning designs; analyses scenario through UML Tool; documents teaching strategies in different formats: rtf, xml, php; manages the unit of learning through the design manager; verifies the unit of learning using constraint editor for checking whether a design or a template is a valid xml scheme (*figure 2*). The educator starts the unit of learning to preview a design in Run Time Environment (RTE).

Performance support subSystem provides end-to-end support of design processes from idea formation through to a complete Unit of Learning (UoL), incorporating communications, assessment, simulations, multimedia content and other parts of an eLearning experience (*figure 3*). The performance support

subSystem is the primary delivery mechanism that provides assistance to the learning designer through some components using Web 2.0 technologies as WikiLDGlossary, Blogs, and Search engine. The first component in gaining proficiency is training through best practice examination, performing learning activities in Automating subsystem and passing the tests. The second component is support with structured information and instructions. The third component of proficiency is experience that is gained over time when a user is faced with various situations that may occur during the unit of learning's design.

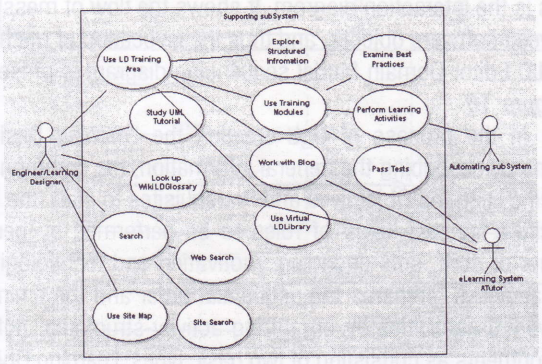

Figure 3. Use Case Diagram of the Supporting subSystem

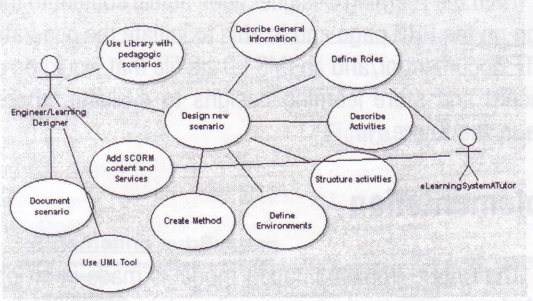

Figure 4. Use Case diagram of Unit of Learning building

From the Learning Designer view point the main activity is the unit of learning creation and this is the reason to examine it in details below. The Use Case diagram in Figure 4 presents the conceptual strategy for building the unit of learning through library pedagogic scenario or designing new scenario, analysis of this scenario at each step using UML tool, adding SCORM content and services to design and document scenario.

The diagram presented on *figure 5* is used for better illustration of the design phase. It consists of two main steps: firstly setting up the components (the unit of learning general information description, role definition, activity description, construction of activity structures, definition of environments), then orchestrate their interplay that means method creation.

Object-oriented Analysis and Design

The Object-oriented Analysis and Design (OOAD) are applied to the CALDS in multiple iterations; the first iteration is for some core functions. Further iterations expand the functionality of the system [14].

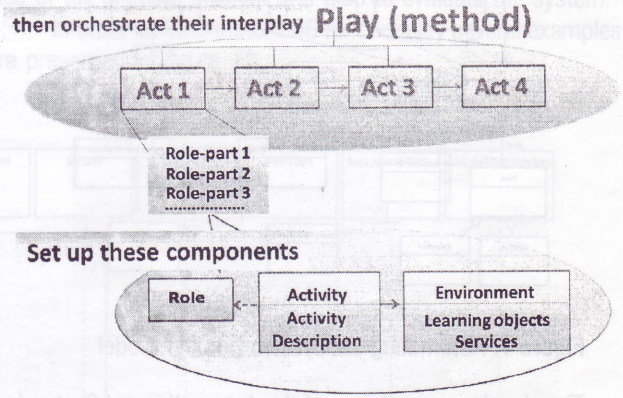

Figure 5. Unit of Learning design

According to Object-oriented Analysis (OOA) some domain models are created: CALSD domain model, Automating subSystem domain model, Supporting subSystem domain model and LD Editor Domain Model. Through the time of Object-oriented Design (OOD) the dynamic behavior of the system is built and in this paper only the LD Editor sequencing diagram is shown.

Domain Model Definition

The first domain model illustrates the conceptual classes in a CALDS domain (*figure 6*); it is the most important artifact to create during OOA. The result is illustrated in a set of diagrams that show the domain concepts or objects. The model illustrates the noteworthy concepts Learning Designer, Automating subSystem, Supporting subSystem, PreviewRTE, UoLLibrary, UoL, UMLTool, TestTool (RELOAD), Role, and eLearningsystem ATutor with their associations.

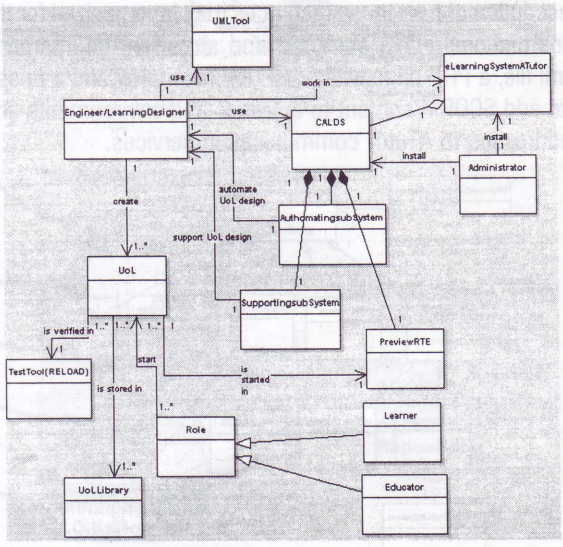

Figure 6. CALDS Domain Model

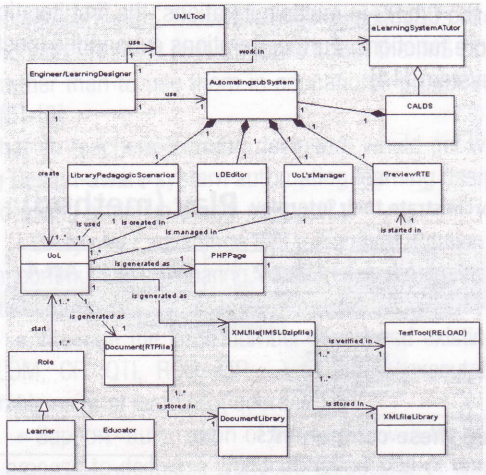

Figure 7. Automating subSystem Domain Model

The visual representation of the Automating subSystem's conceptual classes is presented in *figure 7* and consists of Engineer/LearningDesigner, CALDS, AutomatingsubSystem, PreviewRTE, LibraryPedagogicScenario, LDEditor, UoL'sManager, UoL, UMLTool, PHPPage, XMLfile(IMSLDzipfile), Document(RTFfile), TestTool (RELOAD), Role, XMLfileLibrary, DocumentLibrary, and eLearningsystemATutor.

The identification of conceptual classes in the Supporting subSystem domain is presented in *figure 8* and includes the following concepts: Engineer/LearningDesigner, LDTrainingArea, UMLTutorial, WikiLDGlossary, SearchEngine, SiteMap, Blog, InformationPage, TrainingModule, VirtualLDLibrary, AutomatingsubSystem, and eLearningSystemATutor.

The LD Editor Domain Model is examined in detail as the most important part in the unit of learning creation process (*figure 9*). The LD Editor is comprised of two parts: a Processor that generates a UoL on the basis of Learning Designer interactions and a DocumentingPostProcessor that generates different document formats of the UoL: an RTF file with full documentation of the lesson plans, a reusable IMS LD Content Package with an instructional content, resources for a given learning activity and the description of how those resources may be organized for the best instructional effect compiled and stored in the imsmanifest.xml file, a PHP page with a full UoL taxonomy and a possibility to add SCORM content to activities and environments and URL addresses to ATutor communication services.

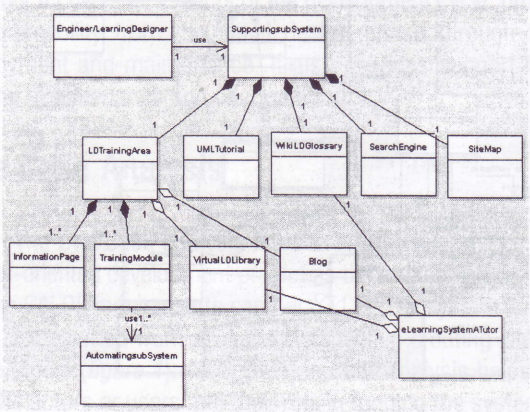

Figure 8. Supporting subSystem Domain Model

The generated UoL is composed of some objects: General Information, Role, Activity, Activity Structure, Environment, and Method. LearningDesigner designs UoL through the LD Editor and by using the Library with pedagogic scenarios and at the same time he analyses concrete educational problems constructing UML diagrams.

Interaction Diagrams Definition

OOD is concerned with defining software objects and their collaborations. A common notation to illustrate these collaborations is the interaction diagram. It shows the flow of messages between software objects, and thus the invocation of methods. The LD Editor Domain Model sequencing diagram is presented in *figure 10*.

In the process of UoL creation the Learning Designer begins by describing the general information, the common and specific Learning Objectives and Prerequisites to start the UoL, specifies the two kinds of roles to be performed by people: learner or staff, sets engaging, motivating and technologically rich activities, prepares the sequential order and the timing of the various activities by organizing activity-structures, defines environments as consisting of two basic types: Learning Objects which would typically be a URL to external content, tools or tests with optional metadata and services provided within the ATutor eLearning system that is available at runtime.

Then the Learning Designer adds actual content to the UoL design on the PHP page and is able to initiate the generation of an RTF document or/and an IMS LD zip file that would be useful to record and store learning designs to evaluate, share and reuse in the future.

Implementation

The CALDS implementation model is made in an object-oriented programming language and contains the artifacts such as source code, database definitions, interface definitions, PHP/XML/HTML pages with dialogs and controls which form part of the proposed system, and architecture. Detailed designs and constructions of each CALDS component (user interface and programmes) have been done, then the integration of the code and user interface in the eLearning platform ATutor. The implementation is largely dependent on the architectural choices

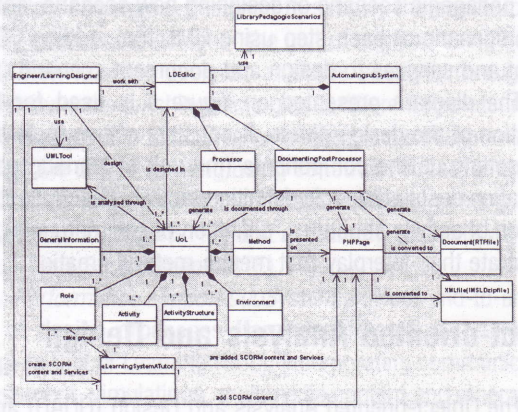

Figure 9. LD Editor Domain Model

middleware and on the used web e-learning environment. In Figure 11 the built CALDS technical architecture is presented.

To gather information about the CALDS with respect to the eLearning context in which it is intended to operate a technical examination in the form of Software Testing is performed. This process is used to help with the identification of the correctness completeness security, and quality of the developed CALDS. By means of alpha testing the system is simulated by potential users and an independent test team at the developers' site test. In this project the Learning Design Tool RELOAD is used for testing the CALDS output product standard format — to check the integrity of the UoL and parse the resources for dependent files. Software prototyping is applied to create a model of the full-featured CALDS, which is used to let the users have an initial idea of the ultimate product and also to evaluate the system.

In order to illustrate CALDS functionality several examples are presented in *figure 12*.

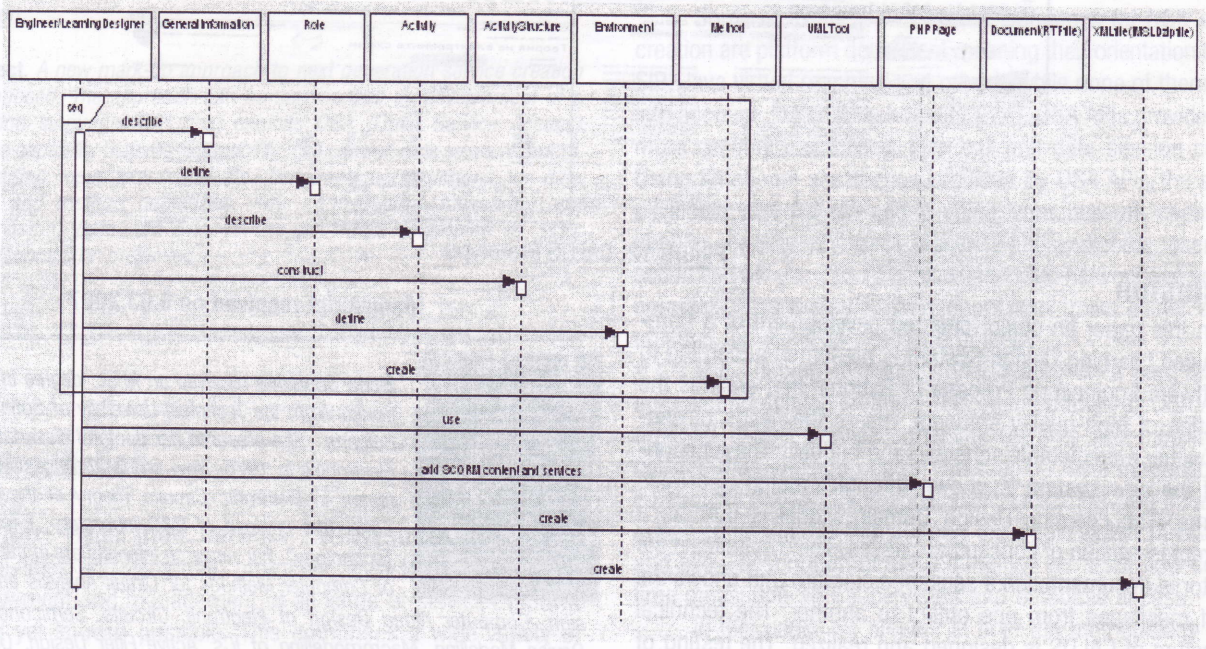

Figure 10. LD Editor Domain Model Sequencing Diagram

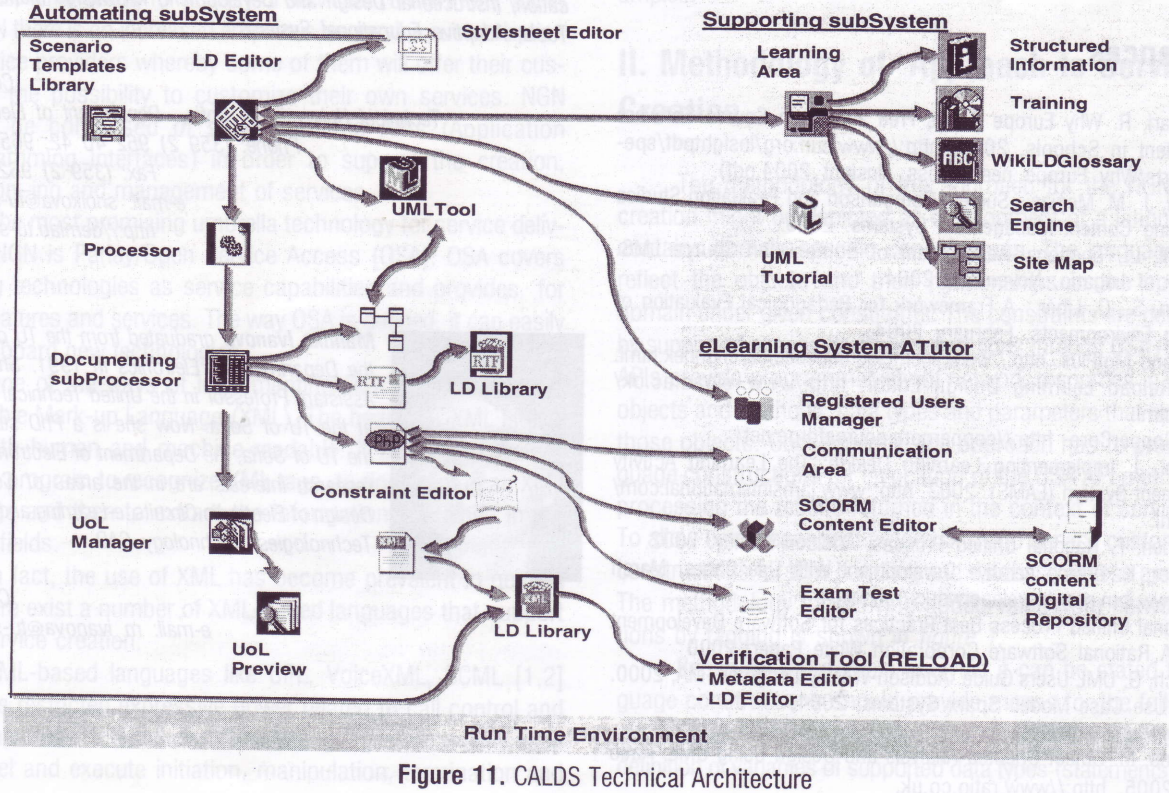

Figure 11. CALDS Technical Architecture

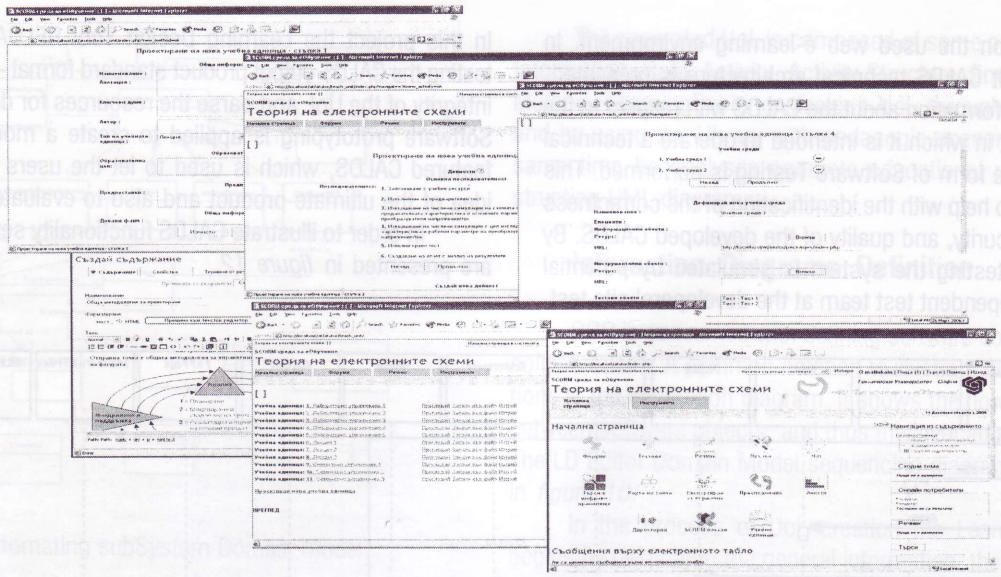

Figure 12. CALDS Prototype

Conclusion

In this paper an object-oriented development of a Computer-Aided Learning Design System is presented according to the Software Engineering Process of Rational that ensures the production of high-quality software, meets the needs of end-users, within a predictable schedule and budget. The functionality of the new system is shown through created Use Case Diagrams. The Learning Design Domain Model is built as a visual representation of conceptual classes. The dynamics in the LD Editor is documented in a sequence diagram that shows the flow of messages from one object to another. The technical architecture of CALDS is designed and realized. The testing of the UoL standard format is performed using the verification tool RELOAD. CALDS evaluation is made on the basis of prototype models.

References

1. Vuorikari, R. Why Europe Needs Free and Open Source Software and Content in Schools. 2004, http://www.eun.org/insightpdf/special_reports/Why_Europe_needs_foss_ Insight_2004.pdf).
2. Itmazi, J., M. Megias. Survey: Comparison and Evaluation Studies on Learning Content Management Systems. 2005.
3. Botturil, L. Functional Assessment of Some Open Source LMS. University of Lugano, November, 2004.
4. Britain, S., O. Liber. A Framework for Pedagogical Evaluation of eLearning Environments. February 2004.
5. Mot and MotPlus. http://www.licef.teluq.uquebec.ca/engndex.html.
6. The Reload Learning Design Editor. http:/www.reload.ac.uk/ldeditor.html.
7. The CopperCore. http://coppercore.sourceforge.net/.
8. Dalziel, J. Implementing Learning Design: The Learning Activity Management System (LAMS). 2003, http: www.lamsinternational.com/index.html.
9. Kratchen, F. Rational Unified Process. Addison-Wesley, 2002.
10. Bittner, K. Driving Iterative Development With Use Cases. March 2006, www.ibm.com/developerworks/rational.
11. Rational Unified Process Best Practices for Software Development Teams. A Rational Software Corporation White Paper, 2000
12. Booch, G. UML Users'Guide. Addison-Wesley-Longman, MA, 2000.
13. The Use Case Model. Sparx Systems, 2004, http://www.sparxsystems.com.au.
14. Collins, M. Object Oriented Analysis and Design Using UML. Ratio Group, 2005, http://www.ratio.co.uk.

Manuscript received on 9.03.2007

Elena Shoikova *received an M.Sc. degree in Electronics from the Technical University of Sofia (TU), Bulgaria. She prepared her doctoral dissertation in Electronics in 1975. Now she is an Associate Professor in Electronic Circuits Theory at the TU of Sofia. She is Head of R&D laboratory „E-learning Technology". Her research interests are in the areas of Computer Methods for Circuit Analysis and Design, Computer Aided Design of Electronic Circuits, Semiconductor Device Modelling, Macromodelling of ICs, Active Filter Design, Object-Oriented Software Engineering, Applying OOAD and a Unified Development Process for software system creation, Professional Standards for Learning Technology, Database Systems with Internet and Java Application, Instructional Design and Development, Knowledge Management Tools, Adaptive Educational Systems.*

Contacts:
Technical University of Sofia, Department of Electronics
Phone: (359 2) 962 40 42; 965 21 40
Fax: (359 2) 962 40 49
e-mail: shoikova@tu-sofia.bg
http://demlab.tu-sofia.bg/

Malinka Ivanova *graduated from the TU of Sofia, the Department of Eletronics in 1991. She is an Assistant Professor in the United Technical College at the TU of Sofia. Now she is a PhD student in the TU of Sofia, the Department of Electronics. Her research interests are in the areas of Computer Design of Electronic Circuits, e-Learning and Internet Technologies, Technology CAD.*

Contacts:
e-mail: m_ivanova@tu-sofia.bg