UNIFIED PROCESS FOR DEVELOPING E-LEARNING SYSTEM

Wahiba Ben abdessalem, doctor in computer science, professor in High Institute of management of Tunis

And

Sami Karàa, distance learning, researcher engineer in National School of Engineers of Tunis
UNIFIED PROCESS FOR DEVELOPING E-LEARNING SYSTEM

Abstract

Learning Management Systems (LMS) or e-learning systems organize and provide training using new technologies. They represent a solution to the growing need for training in many institutions and companies. Many design models have been proposed. However, there is lack of a common software development process. This causes difficulty in evaluating and controlling the product during the development stages. There are norms for LMS (e.g. LOM: Learning Object Metadata) and standards (e.g. SCORM), giving guidelines and frameworks for pedagogical engineering. We emphasize some aspects of modeling (Educational design patterns), using pedagogical concepts to model learning units or learning scenarios, etc. To produce quality software, that meet the needs of users in predictable time and cost, we need a software development process.

The Unified Process (UP) is a software development process using UML. It combines several proven practices from a long experience. It has become standard supported by most important experts in software development.

This paper presents a methodological approach based on the PU in order to achieve an LMS. This process allows to develop iteratively, to handle requirements, to employ component based architectures, to visually model software thanks to UML, to constantly verify software quality, and to control changes during the different phases of LMS development process. This process also shows how elements: workers (essential skills), workflow (essential tasks), and artifacts are defined and applied on a LMS project lifecycle. We show, in addition, how this process can integrate standards, norms, and other aspects of Educational modeling.

Key words: LMS, e-learning, Unified process, UML.

ملخص

منهج موحد لوضع نظام تعليم إلكتروني

(LOM: Learning Object Metadata) وتوجد قواعد مثل (SCORM) وتتمثل بعض جوانب النمذجة (تصميم آثار تعليمية)، وذلك باستخدام المفاهيم التربوية لنمذجة وحدات أو سيناريوهات التعلم وما إلى ذلك. ولكن لإنتاج برامج ذات جودة، التي تلبي احتياجات المستخدمين في مدة مختصرة و
1- Introduction

Learning management systems (LMS) offer an appropriate environment in which an online course can be created and executed. It is a logistics of managing learners, learning activities, and the competency mapping of an organization (Oakes 2002).

The creation of such systems is a hard task, since they are complex systems that incorporate a diversity of technological and pedagogical components.

There is no unified methodology dedicated to LMS. The existing norms, models, and standards provide details about the implementation of a pedagogical framework for e-learning application or give a methodology for the construction of e-learning courses. It is necessary to develop a common methodology describing a development process and taking into account existing tools. The Unified Process (UP) of Object Management Group (OMG) is a generic methodology providing recommendations that can be adopted or extended for different kinds of projects (Monfort et al., 2002), (Pilemalm et al., 2007), (cooper et al., 2006). The UP supports an iterative process that helps specialists in planning and creating their LMS.

The rest of the paper is organized as follows: in the second section we introduce the main concepts of the Unified Process. The third section is interested to the proposed unified Process as a suitable process for a LMS. The last section is the conclusion of this work.

2 - The Unified Process

UP has been progressively recognized, as an efficient process for developing software. UP is viewed as a process framework or a collection of good advices that acknowledges the potential development of the enterprise information system (Kruchten, 2004).

The Unified Process (UP) is a software development process that is iterative and incremental,
Use-Case driven, and architecture-centric (Jacobson et al. 1999). In the Unified Process, models of the system (use case model, analysis model, design model, implementation model, and test model) are created using Unified modeling Language (UML). The UP is organized into four phases or iterations (inception, elaboration, construction, and transition), and five activities (requirements capture, analysis, design, implementation, and test). The unified process makes part of an iterative circle, involving the refinement of all the activities. The iterative approach lets us to take into account changing requirements, to reduce risks in advance, etc. The UP proposes six phases, each phase takes into account a number of use cases to analyze, design, implement, and test:

Inception phase: allows understanding what to build, identifying the system functionalities, and determining a primary and possible architecture. The objective for this first phase is to decide what needs (use cases) to be built early in the process.

Elaboration phase: has as an objective to get more details about the requirements, analysis, design, and implementation, and test. Analysis activity is the most important in this phase. This phase permits some stability of the system architecture.

Construction phase: this phase is incremented by additional use cases. Design, implementation, and test are the largest and main activities in this phase. The product is developed completely. The system is then ready to transition to the users.

Transition phase: in this phase the deployment of the beta version is completed. The team ensures and validates that the user expectations are met.

3. The Unified Process a common process for LMS

In this section we present LMS unified process, a common methodology for LMS building. The process we propose takes its characteristics from the Unified Process, one of the most well-known Software Development methodologies. The LMS Unified Process can be seen as an instance of UP. Like the UP it is use-case driven (user interactions correspond to use cases that denote all aspects of the LMS), architecture-centric (the architecture is a key ingredient in a LMS, that defines the different technologies, the norms and standards, the platforms interactions, the reusability, the execution environment, etc.), and iterative and incremental nature: each activity is repeated in each phase during the LMS development, but also incremental, since at each phase, use cases are further detailed and extended.

Following the UP, we have phases, and activities. Each phase (inception, elaboration, construction, and transition) can be further subdivided into iterations. In every iteration, five activities happen: requirements capture, analysis, design, implementation, and test. Each activity has three dimension characteristics: workflows (list of sub activities), artifacts (the results of an activity), and workers (the personal or skills participating in the development of
the LMS).

We propose in this work various extensions improving and adapting the actual UP (Figure 1). First, an additional phase is introduced: the evaluation phase is necessary for a LMS. Second, workers are added (teacher), and artifacts are included: description of course content.

![Figure 1: the LMS unified process](image_url)

3.1 Evaluation phase

An evaluation step should be added for a LMS to measure the course’s efficacy and to search for opportunities to improve learner’s on-the-job performance (Fardoun et al., 2009). The generally objective of the LMS evaluation is to investigate, evaluate, and make suggestions to improve the LMS. A committee should be engaged for this purpose. The LMS can be evaluated using various criteria and techniques.

Pop proposes two evaluation forms (POP 2012):

- A general evaluation of the LMS: related to functionality, administration, communication and collaboration, accessibility and usefulness, management of e-learning content and users, tools and technology.
- An evaluation of the Learning Objects related to didactic and pedagogical aspect, metadata, content, multimedia presentation, users, and technology.

Alivizos proposes an evaluation method for web resources under education, didactic, and Functional perspectives (Alivizos et al., 2009).

Alkhattabi Proposes a framework that consists of 14 quality dimensions grouped in three quality factors: intrinsic, contextual representation and accessibility (Alkhattabi et al., 2009).

3.2 The activities
3.2.1 Requirements capture

The first activity the unified process is the requirements capture. The requirements capture is first focused on the tasks that the user will perform by using the system, and the functionalities that will be offered to the users.

In the following, we will describe three dimension characteristics: the workflows, the artifacts, and the workers of the requirement extraction activity.

3.2.1.1 The workflows of the requirements capture activity

The requirements capture is decomposed into the following simpler tasks:

• Understand the system context,
• Definition of functional requirements and non functional requirements,
• Identification of actors and use cases,
• Use cases Prioritization,
• Description of use cases,
• Arrangement of the use case model,
• Elaboration of user interfaces.

A-Understand the system context

When developing a LMS, consider the context and environment in which the system will be functioning is crucial. The starting point is to study the information system, to collect data, and provide needs. Information should be extracted from users, and domain experts. This study may require specific techniques to obtain information. In this step we can express the context by a domain model (a class diagram containing the most important classes related to the domain) or a business model (an activity diagram or a sequence diagram expressing the a first drift of business).

B- Definition of functional requirements and non functional requirements

The requirements determine what the users want. Taking the context into account, the requirements are expressed with functional and non functional (technical) dimensions. However, it is not easy to collect all the needs and to be sure they are correctly and completely expressed in the first phase of the unified process. The stabilization of the requirements is done later, progressively, in each phase of the UP. It is always necessary to involve users (actors) in this step, since the essential purpose of this workflow is to attain an accord between the designers and the final users (Jacobson et al., 1999).

The non functional requirements are defined as Requirements which are not specifically concerned with the functionalities of a system. They place restrictions on the product being developed and the development process, and they specify external constraints that the product
must meet (Glinz 2007). The non functional requirements are related to System quality, reliability, usability, maintainability, and cost and development time (Gross et al., 2001).

The functional requirements are defined as functions or components of a software system. The functions may be data manipulation and processing that define what a system is supposed to achieve (Jacobson et al., 1999).

In e-learning systems, functions are performed by a single user (taking course, managing user account, test evaluation, registration, etc.) or in collaboration (forum, chat, etc.). In collaborative environment, several actions are included, such as: communication, coordination, cooperation, etc. Therefore, it is important to include these collaborative tasks in requirements capture.

C- Identification of actors and use cases

A use case is defined as a sequence of actions related to the task the actor and aims to perform, either directly related to the system or to a certain work task (Kruchten, 2004).

The use cases can be identified based on the functional requirements. A use case can regroup several functional requirements. For the LMS, the principle use cases grouped in packages are basically:

**Package of public activities:** provides general information about the distance learning: the profile of the distance learning, the education plan, the financial offer, the possibility of a limited access to educational resources for few days, etc.

**Package of administrator activities:** provides the functions to update the content: update general information, account management of teachers, tutors, and other personal (create, update, delete and select an account), the administration of forums.

**Package of secretary administration activities:** provides the functions for the administration of the students’ information: the level, credit points, studying plan, obtained certificates, etc. The package contains also an actualization of the tutors in charge of every student, the access to various statistics about students (passed exams, not passed exams).

**Package of teacher activities:** is useful to provide necessary tools for course elaboration, to administrate and organize the course (organizational materials: chapter, sections,) in digital format, to define the examination schedule, the evaluation forms, students ‘activities, to communicate with students by chat, forum, or mail.

**Package of student activities:** offers to the students the possibilities of registration (to have an account), to get courses, to participate to a given activity (assignment, test, self evaluation, multiple choice question, quiz,) during the course, to access to electronic library, to discuss with colleagues and tutors through newsgroup, forum, chat or e-mail.

The principal actors are:
Internet user: can access to general information
Teacher: can access and organize activities, to participate to a forum, etc.
Learner: can access activities to get course, to make self evaluation, etc.
Administrator: has the responsibility of forum administration and account management (to add a user we create an account with principal information, login, and password).
Secretary: can accede to students’ information: general information, credit points, passed exams, obtained certificates, etc. and also organizes the tutors, etc.
Tutor: can guide the students in their learning and give assistance when students have difficulties about one course.

These roles are not exclusive; an actor can have a combination of roles. For example, a student can be a learner and an evaluator (for self evaluation); a teacher can be also a tutor.

The use cases and the actors can be represented by a use case diagram (figure 2):

In a learning environment we may incorporate a variety of synchronous and asynchronous communication modes (e-mail, whiteboards, discussion boards, Voice-Over-IP, etc.)

D- use cases Prioritization

It is necessary to prioritize use cases during requirements capture to help decide which use cases should be developed on during early iterations, and which ones can be deferred. Every use case should have an order stating its priority in the development process (Table 1):
<table>
<thead>
<tr>
<th>Use case</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display general information</td>
<td>1</td>
</tr>
<tr>
<td>Course organization</td>
<td>2</td>
</tr>
<tr>
<td>Account Management</td>
<td>3</td>
</tr>
<tr>
<td>Connection</td>
<td>3</td>
</tr>
<tr>
<td>Registration</td>
<td>3</td>
</tr>
<tr>
<td>Course taking</td>
<td>5</td>
</tr>
<tr>
<td>Evaluation</td>
<td>5</td>
</tr>
<tr>
<td>Tutor affectation</td>
<td>6</td>
</tr>
<tr>
<td>Tutoring</td>
<td>7</td>
</tr>
<tr>
<td>Forum administration</td>
<td>8</td>
</tr>
<tr>
<td>Chat</td>
<td>9</td>
</tr>
<tr>
<td>Forum</td>
<td>9</td>
</tr>
<tr>
<td>Newsgroup</td>
<td>9</td>
</tr>
<tr>
<td>Display student information</td>
<td>10</td>
</tr>
<tr>
<td>Statistic Access</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 1: use case priorities

**E- Description of uses cases**

The use case description is habitually defined via an appropriate template with detailed descriptions such as the pre and post-conditions, the different scenarios, the exceptions, the extensions. To describe the use cases in the e-learning domain, we can use standards. A common practice in e-learning is the use of languages to describe the different aspects of a learning scenario (e.g. content, activities, participants, etc). IMS standardization efforts are good examples of this
trend (Friesen, 2005).

- **F- arrangement of the use case model**
  In this step the relations of generalization, and dependencies: “extension” and “inclusion” are revealed between the use cases.

- **G- Elaboration of user interfaces**
  One of the main problems when creating a LMS is the construction of its user interfaces. A good user interface is that allows users to exploit it easily. Attractive interfaces make educational environments appealing for use.

3.2.1.2 Workers of the requirements capture activity
The workers represent the needed skills. In the requirements capture activity they are represented in the unified process by:

**Analyst:** responsible for assembling requirements and documenting them.

**Architect:** responsible of the software architecture. In this activity, the architecture is concerned mainly by the structure of uses cases.

**Use case specifier:** responsible of the description of use cases using natural language, UML diagrams, etc. He should collaborate closely with the users of the system.

**User interface designer:** responsible for designing user interfaces.

For a LMS other workers should be present, the teacher for example participates in the specification of uses cases principally those related to the course elaboration.

3.2.1.3 Artifacts of the requirements capture activity
The artifacts are the result of one task. In the requirements capture activity, the artifacts are: use cases, actors, use case model, architecture description, users’ interfaces, and a glossary.

3.2.2 Analysis
Analysis takes into account requirements capture and starts designing simple and first UML diagrams. It gives a general architecture.

3.2.2.1 The workflows of the analysis activity
The analysis activity is decomposed in to the following tasks:

- **A- Architecture analysis**
  This step concerns the identification of analysis packages and the key classes included in these packages.

- **B- Use cases analysis**
  This step concerns the identification of analysis classes that participate in the realization of a
given use case. The use case analysis is represented by UML diagrams: a static diagram: class diagram in which the classes not contain attributes and operations, and a dynamic diagram: communication diagram or sequence diagram.

- **C- Class analysis**
  The purpose in this stage is to identify the principle attributes and operations of each class, and to have an idea about the role of each class.

- **D- Package analysis**
  The relations of dependency between packages are identified. An actualization of the content of packages is done to ensure that each package contains its proper and suitable classes.

### 3.2.2.2 The workers of the analysis activity

In the analysis activity workers are represented by:

**Architect:** responsible for the coherence of the entire analysis model, his role is to ensure that the packages are well identified and contain the proper classes and the dependencies between packages are correct.

**Use case engineer:** responsible of the realization of one use case or more. He has the responsibility of class diagrams and interaction diagrams and the textual description of use cases.

**Component engineer:** has the responsibility of the class analysis and the package analysis.

In this activity the teacher should be present and participate in the explanation and description of learning materials.

### 3.2.2.3 The artifacts of the analysis activity

In the analysis activity, the artifacts are: Analysis classes, class diagram, interaction diagram, packages, and architecture description.

### 3.2.3 Design

General and detailed design refines the analysis model and used concepts related to the architecture in the modeling. The analysis activity refines the project’s needs, during the design activity; the specialists plan the content of courses, and learner’s actions. The planning and the decision made during the design activity will affect the success of the implementation activity.

#### 3.2.3.1 The workflows of the design activity

The design activity is decomposed in to the following tasks:

**Architecture design:** composed essentially of an identification of the entire dispositive necessary for the deployment of the application and represented by a deployment diagram.

The identification of subsystems related to the application describing a PSI model, TCP/IP model, etc. can be useful in this step of the design activity.
Use cases design: is described first by a class diagram in which all classes contain the all attributes and operations. Stereotypes can be used de assign the class nature. For a LMS, that is in this step that we must take care of learning objects and learning scenarios. Learning objects can be represented by a class diagram. In e-learning, metadata (descriptors) can be used to describe the courses and the learning objects according to a standard LOM (Learning Object Metadata) to enable easy access to educational materials (Downes, 2001).

Since learning materials are the most important elements in learning activities, several researches have been interested to the design of learning materials. The design methodologies are mainly based on SCORM (Sharable Content Object Reference Model), which has become a well-liked standard gathering a set of specifications that expresses how to create learning content sharable and reusable. For instance, learning objects should be SCORM conforming and included in the design model (Fardoun et al., 2009).

Learning scenarios can be modeled using an interaction diagram (sequence diagram or communication diagram) to give the details about the interactions between all the classes’ objects.

A personalized learning can be suggested, to adjust the prioritization of learning by managing the student’s historical learning data (Jeong et al., 2012). IMS LD (IMS 2003) is a standard based on the learners activity can be exploited to express pedagogical scenarios that the learner will pursue during a session.

Design patterns, as a form of design principles, can be exploited to facilitate the design activity and provide general framework for communication (zitter et al. 2009). Learning Management Systems have hypermedia content and can take advantage of design patterns in that field (Avgeriou et al. 2003). The utilization of those patterns for the design and development of the interface, navigation and content of LMS, etc. is interesting since there is a significant amount of work already completed in that area.

For example, Delozanne et al (Delozanne et al., 2007) propose a structured set of 17 Design Patterns that aim the design of learning systems that assess learners while solving problems, (Hernández-Leo et al. 2006) use Collaborative Learning Flow Patterns (CLFPs) to represent techniques that are repetitively used by practitioners in the flow organization of learning activities implicated in collaborative learning situations.

- Classes design
  This workflow concerns the actualization of the attributes and operations of the classes, and to guarantee that the relations between classes are correct.

- Subsystems design
  It is necessary in this step to ensure that each subsystem fulfills its mission and to actualize the
contents of all the subsystems and the dependency between them.

3.2.3.2 The workers of the design activity

In the design activity workers are represented by:

The architect: is responsible of the system architecture and the deployment model.

Use case engineer: is responsible of the design activity of use cases.

Component engineer: defines and actualizes operations, attributes, methods, etc.

In this activity also the teacher should be present and participate in the elaboration of learning materials to maximize the effectiveness of learning materials.

3.2.3.3 The artifacts of the design activity

The artifacts are classes, attributes, operations, methods, subsystems, class diagrams, sequence diagrams, etc.

3.2.4 Implementation

Taking into account the design, implementation conducts to code creation, unit test, and integration of all components.

3.2.4.1 The workflows of the implementation activity

The implementation activity is decomposed in to the following tasks:

Architectural implementation: given the architectural design, the subsystems and their components are identified related to development environment (package in Java, project in VB, directory in C++).

Use case implementation: This activity starts with component identification and then their implementation related to the development environment such as programming language, data base management system, operating system.

System integration: since the system is constructed incrementally, an integration plan is done taking into account the precedent implementations.

Subsystems implementation: given the architectural implementation, the subsystems are implemented related to development environment.

Unit tests: allow to detect programming errors, and to verify that each component runs correctly.

3.2.4.2 The workers of the implementation activity

In the implementation activity workers are represented by:

Architect: responsible of the integrity, the coherence, and the accuracy of the implementation model.

Component engineer: responsible of the implementation of subsystems and their components.

System integrator: plans the necessary developments in every iteration and the integration of implemented uses cases.
3.2.4.3 The artifacts of the implementation activity

The artifacts regroup the components (table, files, library...), subsystems, integration plan.

3.2.5 Test

Numerous tests must be performed. The first test is to check if users’ requirements are correctly implemented and fit the project and the user’s needs. The integration test allows checking that all components work correctly together.

It is important to be sure that the courses are dispensed efficiently and successfully to the learners.

3.2.5.1 The workflows of the test activity

The test activity is decomposed in to the following tasks:

- **Test plan:** the test strategy is described, précising the needed tests, the tests order, the needs of personal, etc.
- **Design of tests:** concerns the description of the tests (method, condition of realization) in every iteration. UML diagrams can be used for this purpose.
- **Implementation of tests:** Tests can be automated. In this case software can accelerate tests and provide better results than those done manually.
- **Integration tests:** the purpose is to test that components interact correctly.
- **System tests:** the objective is to test that the whole system works correctly.
- **Evaluation of tests:** it is a fiability measure of the tests. Other tests can be added if actual tests are judged insufficient.

3.2.5.2 The workers of the test activity

In the test activity workers are represented by:

- **Test engineer:** he guarantees the integrity of test model, plans, designs tests, and describes test procedures.
- **Component engineer:** responsible of tests components and their potential automation.
- **Integration tester:** responsible of test integration in each iteration.
- **System tester:** responsible for testing the system in during a given iteration.

3.2.5.3 The artifacts of the test activity

The artifacts are composed with test cases, test procedure, test component, test plan, bugs, and test evaluation,
4- Conclusions

Even though initially the UP is used for object oriented software development, it has shown to be useful for other kind of projects like a LMS. In this paper, we have extended the UP with additional phase and additional workers. We describe in this paper a methodological approach based on the unified process, adjustable for a LMS. This process allows developing iteratively through the different phases (inception, elaboration, construction, transition, and evaluation). The LMS development process shows furthermore how activities (requirements capture, analysis, design, implementation, and test) are defined and applied on a LMS project lifecycle. We give details about workers and artifacts, implicated in every activity.
References


Oakes, K. (2002). E-learning: LCMS, LMS— They’re not Just Acronyms But Powerful


Zitter, I. Kinkhorst, G. Simons, RG. Cate, OT. (2009). In search of common ground: A task conceptualization to facilitate the design of (e) learning environments with design patterns. Computers in Human Behavior. Volume 25, Issue 5, pp. 999-1009.