A SOFTWARE PRODUCT LINE FOR E-LEARNING APPLICATIONS

Amina GUENDOUZ
CS Department
Saad Dahlab University
Blida, Algeria
guendouz.amina@yahoo.fr

Djamal BENNOUAR
CS Department
Akli Mohand OulHadj University
Bouira, Algeria
dbennouar@gmail.com

Abstract: *As online education becomes a basic need for organizations, a variety of Learning Management Systems is proposed on the market. However, available systems do not satisfy all the needs of the different institutions, which push them to develop their own systems. In this paper, we propose the use of a software product line approach for the development of e-Learning applications to satisfy the variable needs of customers, improve software quality and decrease cost and time of development.*

Keywords: *E-Learning, Software Product Line, reusability, variability management.*

1. Introduction

Nowadays, internet knows a spread use in several fields, namely in the education. Taking advantage from the benefit of using internet, organisations seek to provide an efficient and less expensive way of education in terms of time, cost and effort. Remote training, virtual learning, or electronic learning (e-Learning) means the use of (ICT) Information and Communication Technologies in education to improve the process of teaching-learning.

In order to satisfy the needs of the different institutions, various e-Learning applications have been proposed, the most known are Learning Management Systems (LMSs). A learning management system (LMS), also known as Virtual Learning Environments (VLE), is the infrastructure that delivers and manages instructional content, identifies and assesses individual and organizational learning or training goals, tracks the progress towards meeting those goals, and collects and presents data for supervising the learning process of organization as a whole [12].

In spite of their important advantages, LMSs present several limitations. Dalsgaard [3] argues that LMSs are limited to cover only administrative issues, and suggest the necessity to go beyond LMSs in e-Learning to improved interactions between students and instructors. On the other hand, a survey on LMSs that have been carried out for 113 European institutions [9] revealed that a large number of the LMS systems used in Europe are commercial systems developed locally, or self-developed systems built by the institutions, and that only a few commercial systems are used by several institutions, which means that institutions tend to create their own e-Learning systems to answer their specific requirements.

In order to overcome these issues, we suggest the use of Software Product Line (SPL) engineering approach for the development of e-Learning applications. E-Learning applications could be implemented in a variety of settings: for schools and universities to compliment or enhance classroom learning, for Corporations to provide training and certification for their employees, and for Organizations to provide e-learning courses to a larger learners population virtually anywhere in the world.

However, all of these applications share a set of common software elements and differ by some variable parts. So the adoption of a SPL approach in the e-Learning domain seems to be a promising solution. On one side, to overcome the limitations of LMS systems, and on the other side, to provide institutions with e-Learning applications that fit their own requirements. Furthermore, SPL engineering aim to share the development work of a set of product using a common means of production, in order to reduce the production, maintenance and test costs, decreased time to market and improve quality.

In this paper, we show how to build a SPL for e-Learning applications. The remainder of this paper is organized as follows: Section 2 introduces software product line approach. Section 3 shows the different steps of the development of an e-Learning product line. Section 4 comments on related work while Section 5 summarizes the paper and outlines future work.

2. Software Product Line

A software product line (SPL) is "A set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way" [6]. Software product line approach aims to systematize the reuse throughout all the software development process: from requirements engineering to the final code and test plans. The purpose is to reduce the time and cost of production and to increase the software quality by reusing elements (core assets) which have been already tested and secured. These objectives can be realized by putting in common development artefacts such as requirement documents, design diagrams, architectures, codes (reusable components), procedures of test and maintenance, etc.

Therefore, software product line engineering relies on a fundamental distinction of development for reuse and development with reuse [11], [5]. Domain engineering or "development for reuse" consists in developing the reusable elements (core assets) through the domain analysis, domain design and domain implementation processes. The main outputs of this process are: the identification of the members of the product line (scope), and the extraction of the similarity and variability between them. Application engineering or "development with reuse" consists in developing the final products, using the core assets and the specific requirements expressed by the customers. This process is similar to traditional development process; however, each step is facilitated by the reuse of the outputs of the first process.

However, SPL developers must improve reuse while maintain diversity between products. This could be done by "Variability management".

Variability management is a key activity that usually affects the degree to which a SPL is successful [2]. Variability refers to the ability of an artefact to be configured, customized, extended, or changed for use in a specific context [1]. This variability must be defined, represented, exploited, implemented, evolved, etc. – in one word managed – throughout software product line engineering [11].

When managing variability in a product line, we need to distinguish three main types [11], [5]:

1. Commonality: a software element that is common to all the members of the product line.
2. Variability: a software element that vary from a member to another member of the product line (may be common to some products but not to all).
3. Product-specific: a software element that may be part of only one product at least. Such specialties are often not required by the market per se, but are due to the concerns of individual customers.

During the life-cycle of the product line, a specific variability may change in type. For example, a product-specific characteristic may become variability. On the other hand, a commonality may become a variability as well.

3. Case Study

The software product lines engineering is divided into two complementary processes: domain engineering and application engineering. Figure 1 present the different activities of each process and their outputs.

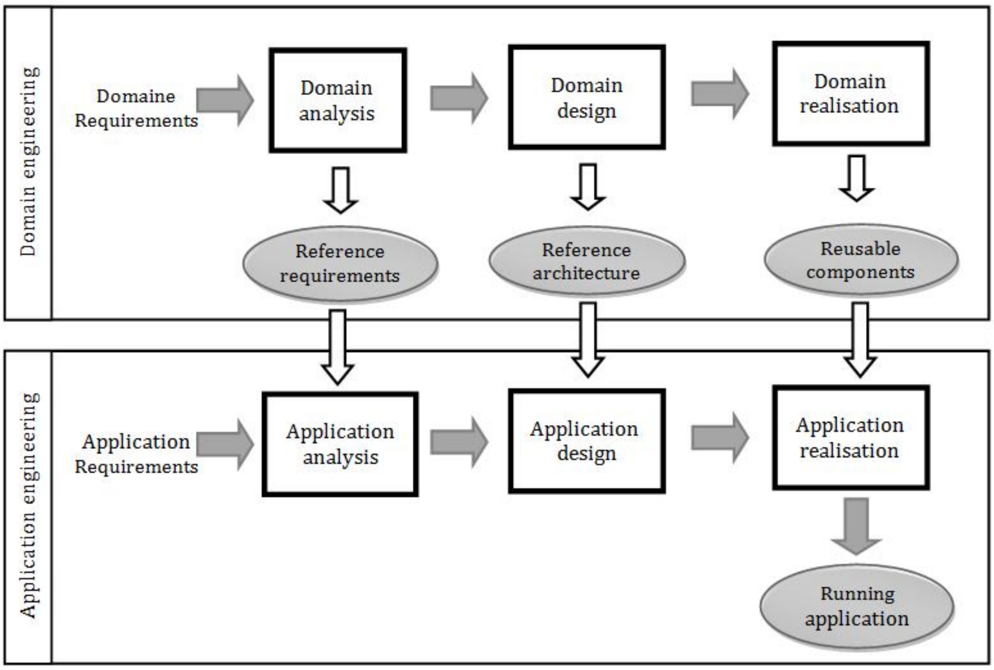


Figure 1: Software product line engineering.

In this section, we show the development process of our e-Learning product line focusing on the first sub-process: domain engineering.

3.1. Domain Engineering

The main purpose of domain engineering is to produce reusable core assets and to provide the effective means that help in using these core assets to build a new product within a product line. A core asset is a reusable artifact or resource that is used in the production of more than one product in a software product line. A core asset may be an architecture, a software component, a domain model, a requirements statement or specification, a document, a plan, a test case, a process description, or any other useful element of a software production process [6].

Domain engineering consists of three activities that are domain analysis, domain design and domain realization (Figure 1).

3.1.1. Domain Analysis

The goal of domain analysis is to define the members of the product line and to extract and document the similarities and variations between them.

In our case, e-Learning product line cover the e-learning systems intended for schools, universities, companies which provide online training to their employees and organizations that supply online courses to learners anywhere in the world.

To extract the common and variable features of our product line, we have used the FORM (Feature-Oriented Reuse Method) approach [4]. FORM use Feature Model to capture commonalities and differences in the scope of the product line. The feature model is generally described by a hierarchy of the set of features of a system or what is called feature tree [5]. Figure 2 and 3 shows a part of the feature model of our case. We do not show the whole feature model for the sake of brevity and space reasons.

The feature model we constructed is divided into two diagrams according to the type of features it includes. Features in the first diagram Figure 2 (capability features), represent the services provided by the system. This diagram shows that the main features of an e-Learning application are "Profile management" and "Course management". However, the application may provide other services such as: "Export content", "Statistics", "Integration with other systems" and so on. The cardinality <0..*> for the feature "Module" means the possibility of adding new sub-features and so the possibility to extend the product line to cover new requirements. A "Course management" must contain at least two features "Content" and

“Enrollment” for a course, but it can include other features such as: “Management of groups”, “Announcement”, “Evaluation”, and “Collaboration tools”.

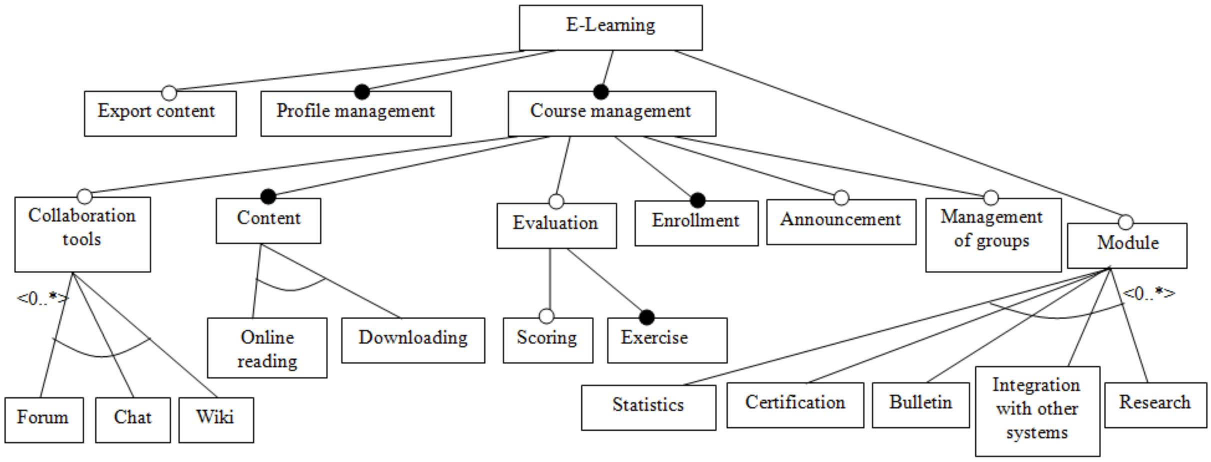
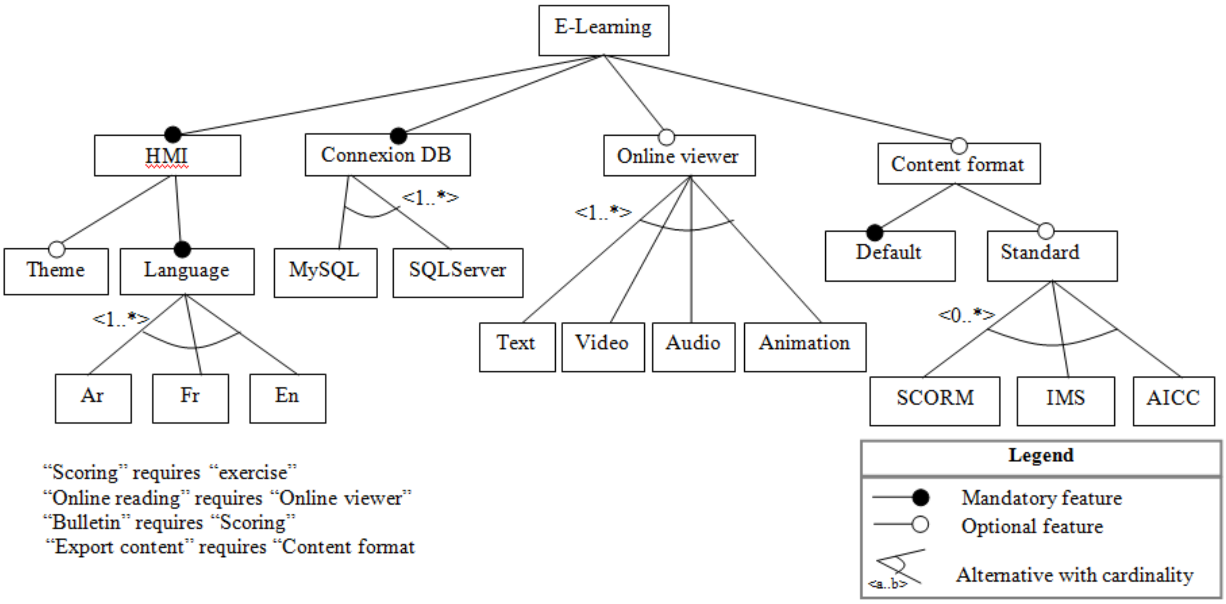


Figure 2: Capability feature diagram for e-Learning product line.

The second diagram Figure 3 represents the implementation features of the system; it means implementation details at lower and more technical levels. An e-Learning application must connect to a data base and supply a HMI (Human Machine Interface). If the application provide “Online reading” of the course's content, this require an “Online viewer” which differ according to the type of the content (text, video, sound or animation). The content of a course can be exported in several formats: default format provided by the system or other standard formats such as: SCORM, IMS or AICC.



“Scoring” requires “exercise”
“Online reading” requires “Online viewer”
“Bulletin” requires “Scoring”
“Export content” requires “Content format

Figure 3: Implementation feature diagram for e-Learning product line.

3.1.2. Domain Design

The purpose of the domain design is to establish the generic software architecture of the product line. Variability identified during domain analysis must be explicitly specified in the product line architecture.

In our case, we have chosen OVM [5] (Orthogonal Variability Model) to represent variability in the design model. OVM consist of a set of Variation Points (VP) and variants (V). A variation point or variability subject shows an aspect of variability within a product line. Variants or variability objects are the different shapes of a variability subject.

To create our Orthogonal Variability Model we have based on the Feature Model constructed in domain analysis phase. Figure 4 shows the variability modeling of the component "content" from the set of components of e-Learning product line. Variability is modeled separately from the component diagram and related to this latter by means of traceability links. In this example, the component "Online viewer" will be implemented in one or more versions: text viewer, video viewer, audio viewer, or animation viewer.

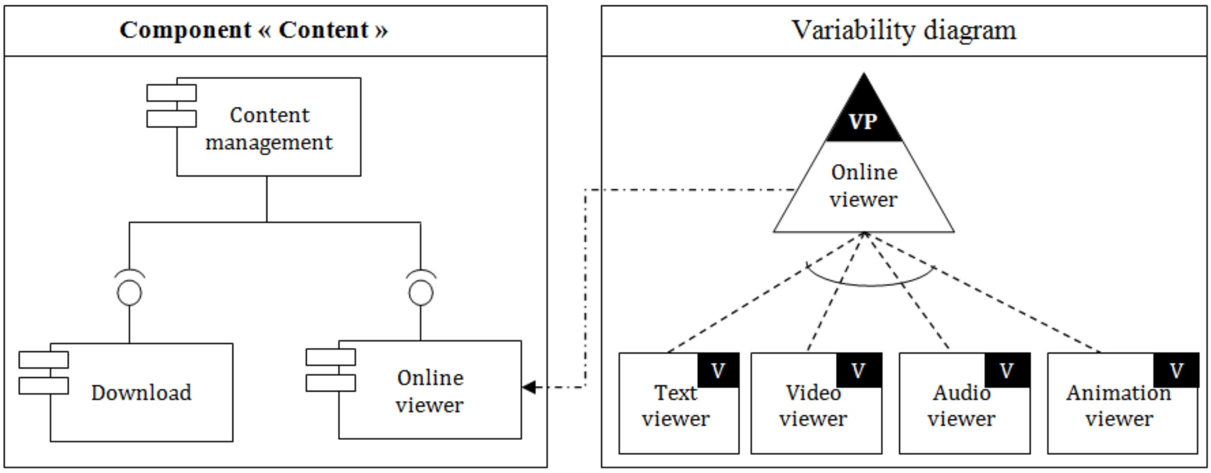


Figure 4: Variability model for "content" component.

3.1.3. Domain Realization

The main object of this step is to create a set of reusable software components. The components that have been identified in the previous step are detailed, planned and implemented to be reused in different contexts. The result of this step is not a running application, but rather, a set of configurable and loosely coupled reusable components.

3.2. Application Engineering

Application engineering consists in developing the final products, using the core assets and customer's specific requirements. This process is also known as "derivation process" of the members of a product line. The results of the domain engineering process contain variations, so the derivation of a particular product needs to make decisions (or choices) about these variation points. Feature model (section 3.1.1) define a decision space for application development.

For each new application, we select the relevant features from the feature diagram according to the specific requirements of this application. The feature model of the particular application is then used to specify which variants must exist in the design model. As a result, we obtain an architecture model without variability, that include only the components of this application selected from the general architecture in addition to the components that implement the specific requirement of this application if they exist. Finally, according to the application's architecture model, we select the components from the base of reusable components obtained in domain realization. In the case where the particular application needs components that have not been predicted by the domain engineering, these application-specific components must be implemented and then assembled with the selected reusable components to create the final running application.

4. Related Work

To the best of our knowledge, SPL approach has not been used to develop eLearning applications. Nevertheless, there is few works that use SPL principles in the domain of eLearning.

SPL was first used in eLearning domain to develop and reuse digital educational content [7], [8]. Pankratius and al proposed the PLANT (Product Lines for Digital Information Products) approach to deal, in a general way, with the issues accoutered in content reuse for eLearning platforms. In this case the reusable elements are a mixture of content and software, since online courses may contain, more than texts, programs and animations.

Another work use SPL approach to develop an auxiliary eLearning application is presented in [10]. Sanchez and al use SPL engineering to develop eLearning Web-miner product line, a family of data-mining applications aiming to assist educators involved in virtual education by extracting and providing useful information that these educators can use to improve the learning-teaching process.

5. Conclusion

In this paper, we proposed the use of a software product line engineering approach to develop e-Learning applications. The aim is to satisfy the variable requirements of institutions, and to benefit from the advantages of SPL engineering, namely: to reduce production, maintenance and test costs, decreased time to market and improve quality. The paper has illustrated the different steps of the development process of an eLearning product line, focusing on domain engineering.

However, more work is needed to improve this e-learning product line, by adding new features to cover a broader scope. It will be also important to define an automatic method of derivation to improve the application engineering process.

References

[1] Bachmann, F., and Clements, P.C., "Variability in Software Product Lines," technical report CMU/SEI, 2005.

[2] Chen, L., Babar, M.A., Nour, A., "Variability Management in Software Product Lines: A Systematic Review," SPLC, San Francisco, California, 2009.

[3] Dalsgaard, C., Social software: E-learning beyond learning management systems, EURODL , 2006.

[4] Kang Kyo C., Kim S., Lee, J., Kim, K., Kim, G. J., Shin, E.: FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures, ,Annals of Software Engineering, vol. 5, pp. 143-168, 1998.

[5] Klaus,P., Böckle,G., and van der Linden,F., *Software Product Line Engineering: Foundations, Principles, and Techniques*, Springer, 2005.

[6] Northrop, L.M., Clements, C.C., "A Framework for Software Product Line Practice," Version 5.0 (online), http://www.sei.cmu.edu/ .

[7] Pankratius, V., *Product Lines for Digital Information Products*, Information Systems, 2007.

[8] Pankratius,V., and Wolffried, S., "A Strategy for Content Reusability with Product Lines Derived from Experience in Online Education", International Conference on Software Engineering (ICSE), USA, 2005.

[9] Paulsen, M.F., "Experiences with Learning Management Systems in 113 European Institutions," Educational Technology & Society, 6 (4), 134-148, 2003.

[10] Sanchez ,P., Diego, G., and Marta, Z., "Software Product Line Engineering for e-Learning Applications: A Case Study," 2012 International Symposium on Computers in Education (SIIE 2012), Andorra, 2012.

[11] Van der Linden, F., Schmid, K., Rommes, E., *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*, Springer, 2007.

[12] Watson, R.W., "An Argument for Clarity: What are Learning Management Systems, What are They Not, and What Should They Become?," TechTrends 51, 2007.