# LMSGENERATOR: Multi-Target Learning Management System Generator Based on Generative Programming and Component Engineering

Rachid DEHBI, Mohamed TALEA, Abderrahim TRAGHA
Hassan II University
Faculty of Science Ben M'Sik
MITI Laboratory, MOROCCO
{dehbirac, taleamohamed}@yahoo.fr, agha@univh2m.ac.ma

***ABSTRACT:*** *Generative and component approaches are revolutionizing software development just as automation and componentization revolutionized manufacturing. Key technologies for automating program development are Generative Programming for program synthesis and Component Engineering for modularity. In this paper we show the contribution of these two approaches in the implementation of a multi-target learning management system generator adaptable to different target runtime environment.*

*This article introduces the basics of a new programming method of virtual learning environments. This approach is based on generative programming that integrates the user specifications (abstract models) and technologies desired in order to produce bricks software, then put them together to produce a solution adapted to area and users' needs. This idea is used at different levels in the design and implementation of our system LMSGENERATOR, a software factory for business components and platforms for distance learning.*

*In this work, we present the challenges, the methodology followed and the implementation of our approach: LMSGENERATOR, with these two generation process and We promote the idea that using this approach based on open business models, adaptable, and scalable to obtain open learning environments, adaptable, and scalable. Specifically, thanks to the generative programming, new technologies or new needs can be easily integrated at any time merely by changing the generators.*

**Keyword:** E-learning, Learning Management System (LMS), Model Driven Engineering (MDE), Model Driven Architecture (MDA), Transformation Approach, Generation, Integration, Process, Business Model, Business Components, Component-Based Approach

**Received:** 2 August 2012, Revised 1 October 2012, Accepted 4 October 2012

## 1. Introduction

Nowadays, information systems have become increasingly complex, a complexity due no only to the richness of content and presentation, but also to the defined entities structure. The device of open and distance learning is an example of a complex and complicated system by its structure, its actors and its operation. Their computerization (eLearning) is a heavy task, complex and complicated that involves multidisciplinary expertise and complementary methods and techniques from several areas: Information

Systems, Software Engineering and pedagogical engineering. However, E-learning is now a reality thanks to the development of computer networks, Internet and especially virtual environments often called platforms eLearning or LMS (Learning Management System).

In the other hand, there have been these recent years, profound changes in computing that influence how to design and develop applications. To meet these changes, it is necessary that the aspects of openness, adaptability, and evolution of applications must be taken into account since the design's phase. So, the design and programming of complex applications, such as LMS, Faced with the growing and changing needs of the area, have to use the standards and take into account the notions of distribution code, deployment, and reuse. Then, it's necessary to change the programming methodologies of these environments.

This article introduces the basics of a new approach to program virtual learning environments. This approach is based on generative programming, from user specifications (abstract models) and the desired technologies, software bricks will be generated and then assembled to produce a complete solution adapted to area and users' needs. This idea is implanted at different levels in the design and implementation of a Learning management system generator baptized LMSGENERATOR. This work shows how the model driven engineering (MDE) in particular the OMG vision of MDE (MDA) and the components engineering can be combined to implement this system.

This article is organized as follows. In section 2, we define key concepts and bases of the Model Driven Engineering, Model-Driven Architecture and Component engineering. Section 3 gives an overview of our approach. In Section 4 we present the software architecture of LMSGENERATOR and its two phases of generations. At the end, we conclude with some issues related to this system.

## 2. MDE, MDA and Component engineering

The software engineering provides constantly new technologies that facilitate the implementation of computer systems [17]. Whatever the approach used (functional, object, components, middleware or services, etc...) The objective is always the same: increasing the quality of information systems and facilitating their implementation on a target runtime environment. This section presents an overview of currently most used approaches in software engineering and that formed the basis for our approach: Model Driven Engineering, Model-Driven Architecture and Component engineering.

### 2.1 MDE

The major drawback of information systems is their increasing complexity and rapid scalability. Deal with this situation, researchers and industries have agreed on the fact that the solution of this problem should result in a rise of models, and a clearer separation between business and technology. This decision has helped to pass the models from their contemplative phase which was reduced to the representation and documentation of computer systems, to a more productive phase which envisages the use of models in the heart of the systems development cycle. That's why the Model Driven Engineering (MDE) has been birth.

The Model Driven Engineering (MDE) is a recent discipline of software engineering that promotes models in first-class entities in software development [18]. It is the subject of great interest from the academic research teams (IDM-Action [19]) and industrial laboratories (Compuware [21], Softeam [23], AndroMDA [20], Xactium [22], etc…).

It is a form of generative engineering, by which all or part of a computer application is generated from templates [10]. In this new perspective, models occupy a prominent place among the artifacts of systems development and in exchange must be sufficiently precise and rich so they can be interpreted or transformed by machines. The process of system development can then be seen as a sequence of transformations [10] of partially ordered models, each transformation taking one or more models as input and producing one or more models as output, until executable artifacts.

### 2.2 MDA

Convinced that the model has become the major paradigm by which the software industry can lift the latch automation development, the OMG (Object Management Group) released in 2000 his vision of IDM: MDA (Model-Driven Architecture) [24, 11]. This is both a proposed architecture and a development approach.

The basic idea of MDA is to separate the functional specification of a system from details of its implementation on a specific platform. For this reason, MDA defines an architecture specification structured in several types of models [9]: Computational Independent Model (CIM), Platform Independent Model (PIM), and Platform Specific Model (PSM), generated from the PIM based on the PDM (Platform Description Model).

The CIM models the requirements of a system, its purpose being to assist in understanding the problem and to establish a common vocabulary for a domain. In UML, the use case diagram is a good candidate to represent a CIM. The PIM, also known as the analysis and design model, is an abstract model independent from any running platform.

The PIM is designed to describe the know-how (skills, expertise) or business knowledge of an organization. Having isolated the business expertise in PIMs, we need either to transform these models into other PIMs for interoperability needs, or to produce PSM models for a specific running platform based on PDMs to improve portability and increase productivity. The PDM modelers the platform on which the system will be executed (component models at different abstraction levels: PHP, EJB, etc…). Specifically, it defines the various features of the platform and specifies how to use them.

**2.3 Component engineering**

The component engineering is considered since the 90s, as a paradigm for development of information systems [12]. The use of this engineering in the early stages of development presents a real interest [13]. Indeed, component engineering essentially addresses two complementary aspects: design for reuse which aims to develop methods and tools to support the production process of reusable components and design by reuse which provides methods and tools to exploit reusable components [15].

The study of these two engineering includes different levels of abstraction and operationalizing that are models, languages , tools and engineering processes. Firstly, we start by setting up a reuse system which is necessary for the creation, enhancement and maintenance of the repository of reusable components. This system sets out the functions of identification, specification, development, validation and organization of the components [14].

Then, from this repository component we designed and developed systems finished by assembling reusable components, like, electronics or mechanical components. Specifically, it consist to design and develop systems from prefabricated components, predesigned and pretested; to reuse these components in other applications; to facilitate their maintenance and evolution; to promote adaptability and configurability in order to produce new features.

## 3. Our Approach

Each year the e-learning's market is in growing development with the new needs [1]. System providers of e-learning must adapt their offerings to the new needs and technological developments. Certainly, the arrival of these tools and on line training systems on the training market are accompanied by a variety of proposals [2, 3, and 4] to meet specific needs but it suffering of many inssuffisence [5 and 6].

Facing the variety of proposals and their insufficiencies, we set a target, the generation of distance learning platforms, adapted to different training organizations and meet all the needs of the area. The criterion of adaptability will also cover the environment of execution.

On the other hand, to implement these solutions (criteria), it is necessary that the aspects of openness, adaptability, and evolution of applications must be taken into account since the design's phase. Therefore, we always have to act on models without forgetting the investment cost caused by the redevelopment and adaptation of existing as well as new business components. Then, these platforms should be generated with a minimal cost and time by adopting the new software engineering practices.

In other words, the engineering of reusable components combined with the model driven architecture approach place development processes of complex systems in a patrimonial perspective whose objective is to build a patrimony, to hoard it, use it, reuse it and develop it. It is this patrimonial perspective that we adopted as an approach to meet our challenges.

In fact, we proposed a development approach [6] that combines the MDA approach to generative programming. It is similar to the software factory concept introduced in [7 and 8]. Except that ours is coupled with the engineering of reusable components.

Our approach accords with the research ideas on Domain Driven Development [7] and on those about models and language for the design and handling of domain reusable components [16]. It is mainly based on the ability to define business models that meet the needs of the area specified in the previous section, and will be processed to generate business components and refined to a specific execution platform. These business components, in turn, will be assembled to generate virtual learning environments.

The methodological concept followed in our research is an approach [6] with three basic steps:

• **Step 1:** It sets up two extensible repositories, one for business models to ensure the business sustainability and the other for the execution platform models to ensure technological evolution. The business models will be reusable, adaptable, flexible, integrating the various phases of the overall process of ODL and representing criteria of adaptability to different user groups, including an educational model that includes different teaching approaches. Similarly, models of execution platform will also be adaptable, extensible and open to technology developments.

• **Step 2:** We apply a set of transformation and refining rules to the business models of step 1, to generate reusable business components and dependent on a target execution platform. These business components will serve to enrich a repository of reusable business components. This step adopts the MDA approach coupled with design for reuse.

• **Step 3:** the Business Components depending on the platform, generated in step 2, are used to generate LMS respecting the rules of the component engineering.

Our approach involves a multidisciplinary expertise and complementary methods and techniques from several areas: Information Systems, Software Engineering and instructional design.

The following figure shows the different facets of our approach.

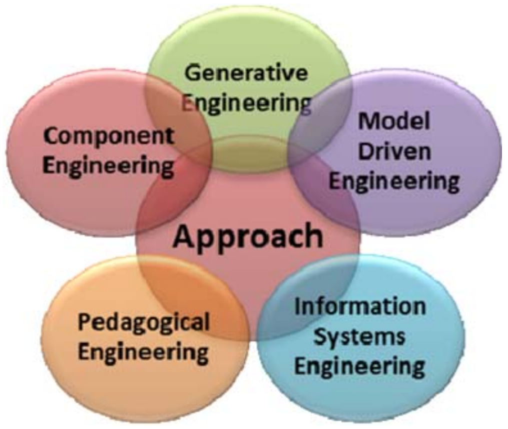

Figure 1. The facets of our approach

This innovative approach in the e-learning field  has required an implementation to become automatic, hence the birth of an original idea that becomes our primary focus and consists of the establishment of a generator, that we name LMSGENERATOR and which the software architecture and its two phases of generation are presented in the following section.

## 4. LMSGENERATOR

LMS Generator is an environment used for integration of business components and generation of other components from model. It aims the generation of platforms for distance learning, executable on multiple execution environments and representing criteria of adaptability to different user categories. Based on multi-layer and interconnected software components (Figure 2), this generator provides these users the ability to:

• Maintain a repository of conceptual components (business model repository) (PIM) durable, customizable, flexible and reusable as well as an extensible repository of platform execution model (PDM).

• Apply transformation, generation and refining rules into models (element of business model repository)in order to generate a business components runtime on the target platforms (using PDM).

• Integrate existing components in its business components repository.

• Describe and maintain models (structure) of learning platform and store them in a descriptive repository for use as needed.

• Generate distance learning environments based on predefined models in the platform descriptive repository (repository of LMS description) or by generating directly a new platform with the ability to save its structure in the dedicated repository.

• Deploy the environments in target execution platforms (on specific application servers and database servers).

LMSGenerator based on an infrastructure using a business model repository and a repository of platform execution models, whose alimentation is currently the subject of our research. This infrastructure is composed of a generator (Business Component Generator), an integrator (Integrator Component) of business components for a specific execution platform and a learning platform builder. The generator is based on model driven architecture approach and permitting, through mechanisms of transformation or merger, to define and generate specific business components (brick), adapted to the construction of a learning management system. Once the repository for business components is supply by the generated or integrated bricks, these bricks will be assembled in the art of reusable components engineering by the learning platform builder to produce a virtual learning environment. The following figure identifies the software components that support the two generation phases of LMSGENERATOR detailed in the following Section.

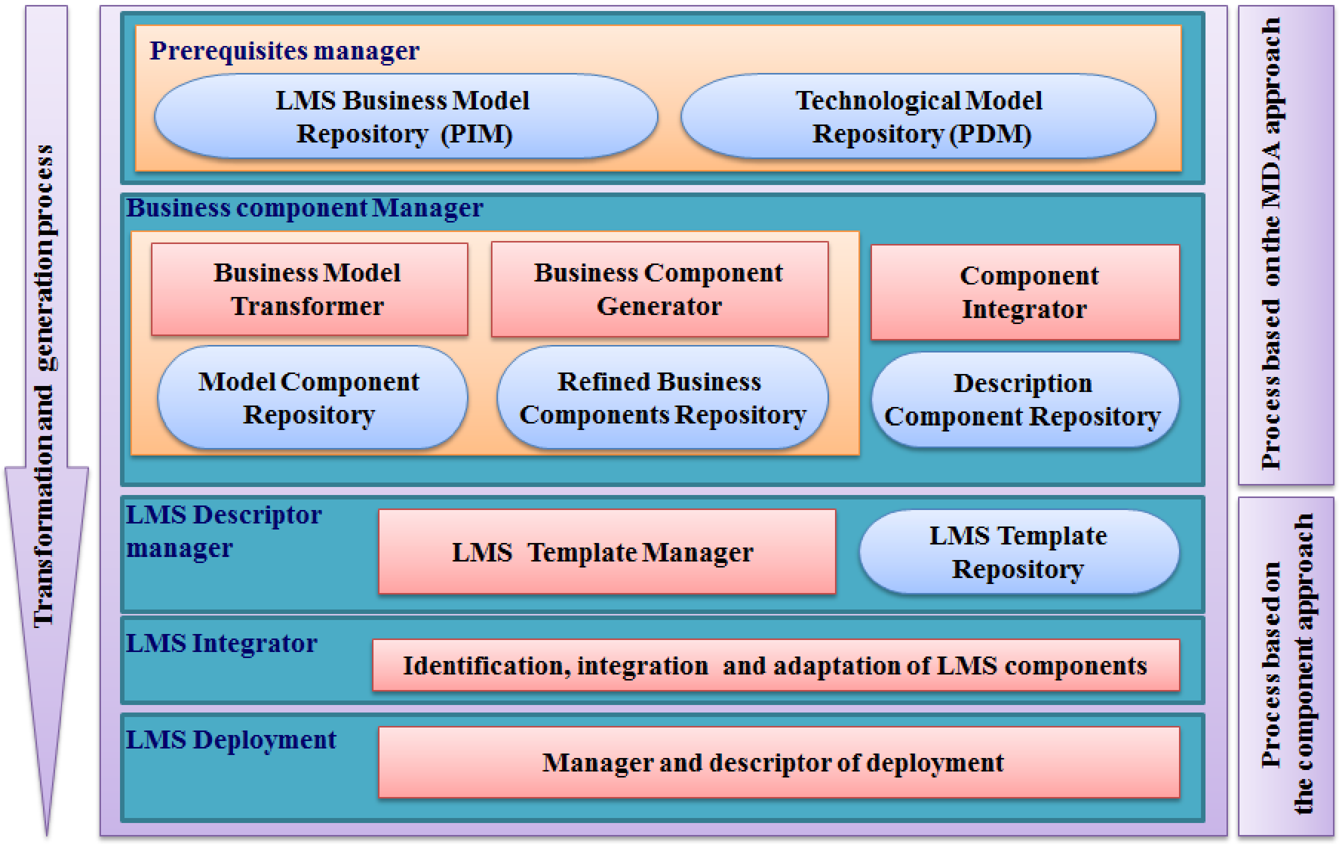
Transformation and generation process
Prerequisites manager
LMS Business Model Repository (PIM)
Technological Model Repository (PDM)
Business component Manager
Business Model Transformer
Business Component Generator
Component Integrator
Model Component Repository
Refined Business Components Repository
Description Component Repository
LMS Descriptor manager
LMS Template Manager
LMS Template Repository
LMS Integrator
Identification, integration and adaptation of LMS components
LMS Deployment
Manager and descriptor of deployment
Process based on the MDA approach
Process based on the component approach

Figure 2. LMSGENERATOR Architecture

**4.1 Generating Process based on MDA**

This phase aims to build LMS business components representing criteria of adaptability and dependent on a target technology (target runtime). It is built on a generation process based on MDA approach supported by two software components of LMSGENERATOR: Business Model Transformer and Business Component Generator.

Starting from a conceptual business model(element of Business Model Repository), we can apply a set of transformation, derivations rules and successive enrichments to produce a refined LMS business component (element of Refined Business Component Repository), ready to be used as a fundamental building block in the construction of our learning platform.
Each business model from our LMS business repository is represented by three sub-models:

• The data model for describing the structure of the component.

• The semantic model to structure and modularize the code of rules, facilitating maintenance and reuse.

• The visual model to define different views of data, independently from visualization means.

These three sub-models should be independent of any application and technology and represented using the UML (Unified

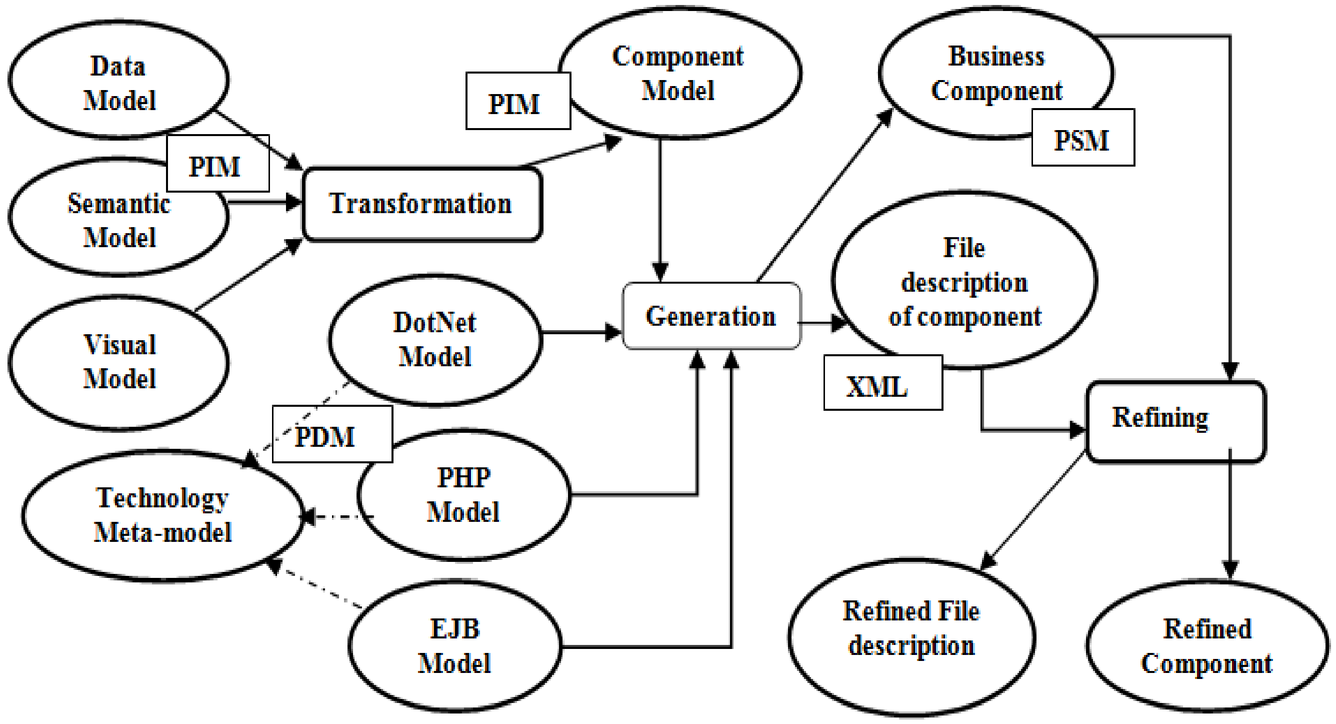

Figure 3. Generation process based on MDA approach of LMSGENERATOR

Modeling Language) formalism: class diagram, sequence diagram, activity diagram and OCL (Object Constraint Language). Indeed a business model is a first class entity for our build process. A business model based on a data model and a model that describes its semantics.

The data model contains a description of the entities involved in the business model while the semantic model describes the interactions and constraints between these entities and their associated behavior in the context of business expertise. The visual representation of the business model is the responsibility of the visualization model. This model which offers different graphical views of business model is considered as a form of representative body of data model, structure, typology and nature of the business model (web service, web site). Thus, with minimal effort, a component model independent from technology can be quickly obtained by a series of transformations, mergers or refinements of the three models mentioned previously (figure 3),This transformation is the charge of Business Model transformer (Figure 2). This component model, as close as possible to the needs of the application, specifying the services provided and required, enter in its turn in a generation process supported by Business Component Generator (Figure 2).

This generation process is based on a set of conceptual models of technology development defined in our repository of technology model (figure 3), in order to generate business components specific to an execution platform as well as their descriptive XML file. This XML file contains a detailed description of the internal components of the business component (web page, class, SQL schema). It also serves to feed the descriptive repository of the components used when building the platform. Once business components are generated, they are refined and ready for use by the process of LMS generation of the second phase. The figure below, present a partial view of the generation process based on MDA approach.

**4.2 Generating Process based on component approach**

The purpose of this phase is the construction and deployment of virtual learning environments (LMS) in a target runtime. It is based on a generation process adopting the component approach. Starting from a repository of LMS business components, generated in the first phase, or fueled by integration of existing components, we can proceed by assembling, adapting and refining to produce each time a new device for learning adapted to the needs of training organizations. In this phase we can also create models of learning management system regardless of the technology, stored in the LMS structure repository for later use by the construction process.

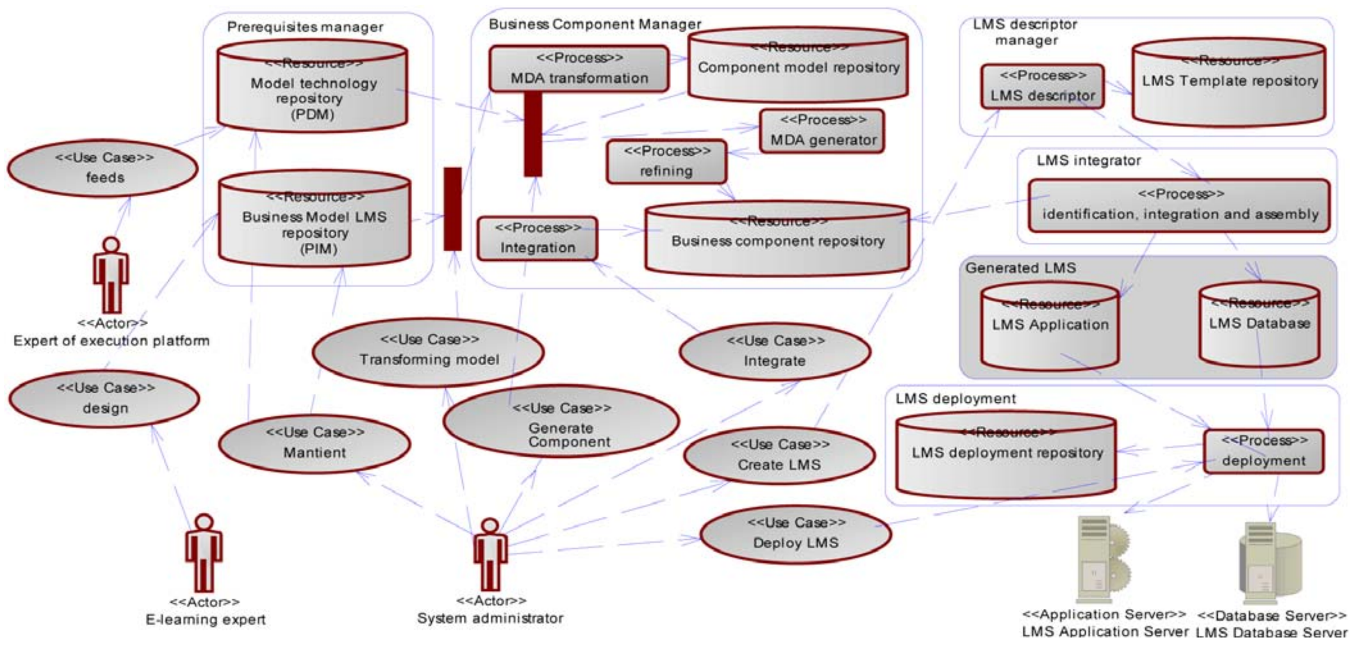
Prerequisites manager
<<Resource>> Model technology repository (PDM)
<<Resource>> Business Model LMS repository (PIM)
Business Component Manager
<<Process>> MDA transformation
<<Resource>> Component model repository
<<Process>> MDA generator
<<Process>> refining
<<Process>> Integration
<<Resource>> Business component repository
LMS descriptor manager
<<Process>> LMS descriptor
<<Resource>> LMS Template repository
LMS integrator
<<Process>> identification, integration and assembly
Generated LMS
<<Resource>> LMS Application
<<Resource>> LMS Database
LMS deployment
<<Resource>> LMS deployment repository
<<Process>> deployment
<<Use Case>> feeds
<<Actor>> Expert of execution platform
<<Use Case>> design
<<Use Case>> Transforming model
<<Use Case>> Mantient
<<Use Case>> Generate Component
<<Use Case>> Integrate
<<Use Case>> Create LMS
<<Use Case>> Deploy LMS
<<Actor>> E-learning expert
<<Actor>> System administrator
<<Application Server>> LMS Application Server
<<Database Server>> LMS Database Server

Figure 4. LMSGENERATOR architecture based on generation process

The construction process of LMS begins once the process of generating business components completed. This process is triggered by the LMSGENERATOR administrator which begins the process of construction by the description of the desired platform. Under the format of an XML file, this description (structure) is a set of functional specifications meeting the descriptions generated bricks of business components repository (phase1). Once completed the step of description, the process proceeds by building identification of the specified components in the structure for generating a LMS learning device.

This process of LMS generation brings these bricks, and then refines, integrates and adapts them to build a device that will be deployed by the deployment manager on the target runtime environments. This process of generation and deployment is supported by three LMSGENERATOR software components: LMS descriptor manager, LMS integrator and LMS deployment. The figure (Figure 3), represent a complete view of the generation process supported by LMSGENERATOR.

## 5. Conclusion

Our approach facilitates the design of Virtual Learning Environments, through their construction by assembling components generated from model or from components directly integrated into the Business Component Repository. It also accelerates their development and deployment by the principle of software reuse, as it facilitates their development by providing a clear separation between specification and implementation components.

Since the models are more perennial than their implementation, we adopted this approach in the will to implement the reuse of properties, cooperation, adaptability and interoperability in the development of on line learning platforms. Reuse reduces cost and time for design, implementation and maintenance. The adaptability makes the environment flexible while cooperation and interoperability makes it open. The main advantage of our approach is the consideration of future developments in technology or domain. These developments are supported by modifying (or creating) generators associated with LMSGENERATOR that will automatically propagate changes across all environments and components products.

We have presented in this work, our approach and an overview of LMSGENERATOR software architecture with these two generation phases. And we have demonstrated the importance of model driven engineering and component engineering in the development of this generator. Future work will include the finalization of this generator, by defining the rules of transformation and generation implemented in these cores, and by feeding the business and technological repository. Also, by defining an adapted component model and a new language based on XML: syntax for describing models and learning platform independently of technology and that will be used in the generation process.

## References

[1] Oubahssi, L., Grandbastien, M. (2006). La e-formation : Comment répondre aux différents besoins de ses acteurs ? Revue ISDM, N° 25- TICE Méditerranée –mai.

[2] Etude comparative technique et pédagogique des plates-formes pour la formation ouverte et à distance, La Direction de la Technologie, sous-direction des technologies éducatives, des technologies de l'information et de la Communication (DT/SDTETIC) du ministère de la recherche et du Fonds Social Européen, Septembre (1999),

[3] Ecoutin, E. (2000). Mise à jour de l'Etude comparative technique et pédagogique des plates-formes pour la formation ouvert et à distance, ORAVEP.

[4] Oubahssi, L., Grandbastien, M., Claës, G. (2003). Analyse Synthétique de cinq plates-formes de formation ouverte et à distance, (rapport interne), p. 46. Janvier.

[5] Oubahssi, L. (2005). Conception de plates-formes logicielles pour la formation à distance, présentant des propriétés d'adaptabilité à différentes catégories d'usagers et d'interopérabilité avec d'autres environnements logiciels, Thèse de doctorat de l'Université René Descartes, Paris V dans le cadre d'A6/OMERIC.

[6] Dehbi, R., Talea, M., Tragha, A. (2012). LMSGENERATOR: A new approach to generate learning management system, *International Conference on Software Engineering*, Databases and Expert Systems (SEDEXS'12), June, p. 14-16.

[7] Czarnecki, K., Vlisssides, J. (2003). Special Track at OOPSLA'03, October. Domain-Driven Development

[8] Courbis, C., Parigot, D. (2003). La programmation générative pour le développement d'applications : les apports pour l'architecture, ICSSEA.

[9] Blanc, X. (2005). MDA en action: Ingénierie logicielle guideé par les modèles, Eyrolles.

[10] Diaw, S., Lbath, R., Coulette, B. (2009). Etat de l'art sur le développement logiciel basé sur les transformations de modèles. TSI - 29. Ingénierie Dirigée par les Modèles.

[11] OMG, MDA, (2001). http://www.omg.org/mda/specs.htm.

[12] Barbier, F., Cauvet, C., Oussalah, M., Rieu, D., Bennasri, S., Souveyet, C. (2002). Composants dans l'ingénierie des Systèmes d'Information: concepts clés et techniques de réutilisation, dans Démarche d'ingénierie de systèmes à base de composants, Actes des deuxièmes assises nationales du GdR I3,

[13] Oussalah, M. (2005). Ingénierie des composants: concepts,techniques et outils, Ouvrage collectif Vuibert.

[14] Khayati, O. (2005). Modèles formels et outils génériques pour la gestion et la recherche de composants, thèse de doctorat à l'institut national polytechnique de Grenoble, Le 17 décembre.

[15] Pujalte, V., Ramadour, P., Cauvet, C. (2004). Recherche de composants réutilisables : une approche centrée sur l'assistance à l'utilisateur, Actes du 22ème congrès Inforsid, Biarritz, p. 211-227, 25-28 mai.

[16] Ramadour P. (2001). Modèles et langage pour la conception et la manipulation de composants réutilisables de domaine, thèse de doctorat à l'Université d'Aix-Marseille III, Le 17 décembre.

[17] Bézivin, J., Blanc, X. (2002). Vers un important changement de paradigme en génie logiciel, *Journal Développeur Référence* http://www.devreference.net/, Juillet, p. 1-9.

[18] Bézivin, J. (2004). Sur les principes de base de l'ingénierie des modèles, RTSI-L'OBJET, 10 (4)145-157.

[19] Action IDM, http://www.actionidm.org.

[20] AndroMDA (2008), available at: http://www.andromda.org/.

[21] OptimalJ - Model-driven development for Java, Electronic Source: Compuware, (2003). http://www.compuware.com/products/optimalj/.

[22] XMF-Mosaic, Electronic Source: http://www.xactium.com.

[23] Softeam, support de formation Objecteering 6/MDA Modeler version 2.0, Janvier (2008).

[24] Soley et al. (2000). MDA (Model-Driven Architecture), White Paper, Draft 3.2, November 27th, OMG Staff Strategy Group.