ON THE WAY TO MODELING: A CASE STUDY IN MANAGING MODEL TRANSFORMATIONS IN MDA

Ahmed A. Saleh, Hazem M. El-Bakry, and Mahmoud A. Atta Alla
Faculty of Computer Science & Information Systems, Mansoura University, EGYPT
E-mail: helbakry20@yahoo.com

Nikos Mastorakis
Technical University of Sofia, BULGARIA

Abstract:
The object technology revolution has allowed the replacement of the procedural refinement paradigm by the more fashionable object composition paradigm. This change has radical effect in software development. The Object Management Group (OMG) is rapidly moving from its previous Object Management Architecture vision (OMA) to the newest Model-Driven Architecture (MDA). This paper shows the importance of the modeling methods as presented by model driven architecture (MDA) and the usefulness of Unified Modeling Language (UML) on the issue of modeling. And The benefit of MDA how it will go along with the development based on software development life cycle (SDLC) process, from analysis and requirement phase through design and implementation stages as employed by object oriented system analysis and design approach using real case study “Modeling e-LMS for Telecom-Egypt training-sector”.

Keywords: e-Learning, OOA&D, UML, PSTN, MDA, XML, CIM, PIM, PSM.

I. Introduction

Today, one of the key standards for Model Driven Development is the Model Driven Architecture (MDA) [24] conceived by the Object Management Group’s (OMG) vision of a model based development. The approach suggests that abstract models, which are platform independent, are transformed in a systematic way to generate deployable, platform-specific implementations. These transformations offer materialization collections of design decisions an analyst is required to make in order to satisfy requirements of the application being developed. The goal of this paper is to show that the same approach can be applied when developing e-LMS for Telecom-Egypt training sector and doing so by using tools either already present on the Market (or in the open-source community). Telecom-Egypt is only PSTN (public switched Telephone Network) company in Egypt that has 55000 employees and 11000000 customers. Telecom-Egypt has a large Training Sector that consist of nine institutes are geographically distributed over Egypt. Hence face a number of problems in getting learning materials and quality trainer. The training sector faces critical issues that are global in perspective. These are the train may be take week or many weeks and the shortage of specialist and engineer make it impossible to leave their work and go to Cairo to take courses & the money that company paid for travel from any branch to training sector. Not only previous two problems face telecom-Egypt training sector, but there other problems arises from traditional training process we explain it in the background study sections. And so the optimal solutions to the previous problems that can solve time & distance problem is E-learning system designed by MDA approach. E-LMS (e-learning Management system) play central role in the web based scenario. It connects learning contents and learners in standardized manner. It manages trainee, learning materials and training events. And this research discusses the e-learning, describe which technology and security may be needed to this system and metrics in orders to improve the evaluation of capability and maturity of e-learning. The development uses object oriented system analysis and design (OOA&D) approach base on model driven architecture (MDA), Unified Modeling Language (UML) is mainly used and extensible markup language (XML). In this research paper we use the Modeling methodology because is the best way to create visualization of the system to be defined. With modeling, complex and real-world systems can be somehow understood, qualities of the system can be predicted and communication with stakeholders (instructor-student-administrations) concerning the key characteristics will be promoted [7].

This paper is organized as follow. In section II, we describe introduction our system and its problems, in section III focuses on OOA&D in section IV and four gives the employment of Model-driven Architecture and the use of Unified Modeling Language (UML) as the documentation modeling language specified by OMG. In section V The way of creating Platform Independent Model (PIM) and means of transformations of PIM into Platform Specific Model (PSM). Section VI explains our case study in Managing Model Transformations in
II. OBJECT ORIENTED SYSTEM ANALYSIS AND DESIGN (OOA&D)

There are several approaches used in system analysis and design. The most two commonly used approaches are: structured system analysis and design (SA&D) and object oriented system analysis and design (OOA&D). Kendall [8] discuss that structured analysis prescribes analyzing and designing software system through functional decomposition. It examines an Information System in terms of the functions it performs and the data it uses and maintains. SA&D identifies the major functions or processes of a system, then breaks or decomposes each function down into its smaller composite steps [8]. On the other-side OOA&D decomposes the system down into objects and examines how these objects act and interrelate. The analyst first identifies the object that comprises the system, then create an object model which groups the objects into classes, and describes each class in terms of its attributes (or data), methods (or functions) and relationships to other classes [9]. Or on other way we can say as Bui M.D. [10] in the past, when modeling was in process, designs are conducted in a top down manner, so the methodology of complexity reduction was called functional decomposition. Combining elementary objects to form a larger object is the bottom-up method and the process is called composition. Although composition/decomposition have been used for a long time in conventional modularization techniques, they are still valid mechanism of complexity reduction in the context of object-orientation. When decomposing, objects identified at each level of decomposition compose a layer at that level. This process thus creates a hierarchy of objects with the most complex object lying on the top of the hierarchy. Elementary objects are found at the bottom layer I will explain these two ways by this example:-

The car system can be broken down into major interacting subsystems, such as the engine, the transmission, the cockpit . . . that are made up of more primitive components. For instance, the engine consists of carburetion and cooling subsystems. At the most detailed level (bottom layer), subsystems are primitive physical component like screws, metal sheets, molded parts . . . The car’s cooling subsystem can be described in terms of radiator, water reservoir, rubber tubing, and channels through the engine block. The car stays alone at the toplemost layer. At intermediate layers, we found engine, transmission, brake, direction, passenger cockpit. At lowest level, a car repair shop can list a computerized inventory of thousands of parts.

Hierarchy and layering technique correspond to aggregation/composition counterpart in object technology. But this is not the only structuring mechanism governing the world of complexity [10]. As applications become more and more sophisticated, they associate many objects coming from various disciplines. Tasks in everyday life are executed with the contribution of many objects which cannot be put inside a hierarchy. In this case, the term collaboration is used to describe such association, and objects are collaborative objects/agents assuming some roles while achieving a common goal. The same object may have multiple roles if it can execute more than one task. If aggregation/composition hierarchy is seen as a vertical mechanism structuring the world, collaboration is the horizontal mechanism. In fact, modern systems call for these two mechanisms simultaneously at various levels. If we consider now the previous car with a conductor inside interacting with the car and looking at the highway, we have both a hierarchical system (the car) and collaborative objects (conductor, car, local geometry of the highway, other vehicles, local meteorological objects like fog or snow, mobile phone, etc.). All these objects cannot be aggregated inside the car hierarchy, so, only collaboration could be the most appropriate description at the driving level. In a normal situation, all the car objects are hidden from the car hierarchy to simplify the discussion on driver reactions, but in case of an accident caused by the driver grasping his mobile phone, the quality of a hidden component, its airbag, will be of highest importance.

OOA&D approach will be employed in development of e-LMS. UML is also the one to be used in MDA. In software development successful information system are subject to frequent evaluation and revision within a framework known as System Development Life-Cycle (SDLC). The e-LMS is a system to be developed hence will abide to System Development Life Cycle (SDLC). SDLC for e-LMS will have:

1. requirement and analysis phase where use-case and conceptual diagrams will be used
2. design phase where design class and component diagrams will be used
3. Implementation and testing phase where coding will be implemented.

III. Towards Model Driven Architecture (MDA)

Modeling: [1] is about recognizing the problem and devising an appropriate approach - one of which might be to apply the UML in one or more areas to help to solve, or at least clarify, the problem. A model is an abstraction of the real thing. Or Modeling: its provides a general skill set that can help the understanding,
communication and resolution of problems [2]. When
you model a system, you abstract away any details that
are irrelevant or potentially confusing. Your model is a
simplification of the real system. A modeling language
can be anything that contains a notation (a way of ex-
pressing the model) and a description of what that nota-
tion means (a meta-model). The development of eLMS
for Telecom-Egypt Training Sector explain in section
six will finally be Modeled by Transform it through
MDA models to reach the XML document that any
developer can implement it by using any programming
language. The creation of eLMS platform specific
model Object Management Group (OMG) is promoting
model driven Architecture (MDA) as a way to develop
systems that more accurately satisfy customer’s needs
and that offer more flexibility in system evolution
(Brown, 2004) [7]. MDA improves requirement capture
and system specification. The UML represent a col-
collection of best engineering practices that have proven suc-
scessful in the modeling of large and complex system.
The UML [4] can help even non-programmer to under-
standing the overall functionality of the system. We
choose this method because the modeling design pat-
terns with UML have several advantages include It quite
naturally to object oriented modeling, Process models
can be communicated more easily in a large number of
people, UML provide a large set of diagram which can
be used to define both structured and behavior of dy-
namic software process. Object oriented modeling sup-
port the earlier phase of process model development
[3].The application of such technologies allows us to
design and implement. Some highly complex and capa-
bile systems. Understanding where systems are being
deployed, who uses them, how new systems need to
integrate with existing systems and what specific busi-
ness tasks they support are the keys to implementing a
successful information system. UML is one among the
technologies specified by OMG as a language to enable
model driven approach. OMG in its UML specification
visualizing, specifying, constructing, and documenting
the artifacts of a software-intensive system. The UML
offers a standard way to write a system’s blueprints,
including conceptual modeling. With the use of UML at
its metamodel level, i.e. logical model level, several
modeling graphics with different viewpoints will be
created as a visual representation of the model. Promote
action research methodology by providing a basis for
understanding the expected platform and promote a
ready-to-use motion.
Finally, MDA helps [25] also software users cope with
two key realities of today’s software environment; mul-
tiple implementation technologies, and the need for
maintenance over extended software lifetimes. MDA
uses OMG modeling standards to create and manipulate
precise, detailed, machine-readable models that repre-
sent application structure and behavior independently of
what programming languages, operating systems, data-
bases or other technological platform being used to im-
plement them. MDA’s basis in freely-available stan-
dards has resulted in a thriving community of tool ven-
dors and open-source tool offerings. Use of standards
allows tools from multiple vendors to be used together
on a single project - a vital feature, since even the larg-
est tool vendor may not provide support for all neces-
sary software platforms.

IV. The Vision of MDA

Using MDA with open source tools makes it possible to
create computer readable models of databases and file
transformations, facilitating auto generation of applica-
tions and significant code reuse. Implementation of sig-
nificant design changes often takes just a few minutes.
The PIM is reused for many purposes Including report
generation, database access, and user documentation.
The vision of MDA is both simple and great. It objec-
tive is to decouple the way that application systems are
defined from the technology they run on. The purpose
of this decoupling is to ensure that the investments made
in building systems can be preserved even when the
underlying technology platforms change. MDA is a
conceptual framework created by the OMG that separ-
ates business-oriented decisions from platform deci-
sions to allow greater flexibility when architecting and
evolving software development and deployment [7]. To
support these principles, the OMG has defined a specific
set of layers and Transformation that provide a concep-
tual framework and vocabulary of MDA. These are
Computation Independent Model (CIM), Platform Inde-
pendent Model (PIM) and Platform Specific Model
(PSM) [12] [7] [13] as shown in fig. 1. The MDA [10]
unifies every step of the development of an application.
It separates clearly Platform Independent Models (PIM)
from one or more Platform Specific Models (PSM). The
PIM is a representation of business functionality and
behavior, undistorted by technology details. In order to
implement a PIM on a specific platform, a tool is used
to generate the PSM from the PIM the tool understands
the target. Technology and knows how to translate the
logical constructs of the PIM into a suitable form for the
chosen platform. These mappings can be complex, and a
given element of the PIM may be mapped to multiple
elements in the PSM. For instance a single business
object of the PIM may be mapped to an SQL DB Table
definition, an EJB Entity Bean and a Remote Interface.
The PSM is also a model, but now at the design level and reflecting the mapping to a target platform. The final stage is to generate from the PSM the code and other technical artifacts (SQL DDL, IDL Interfaces, deployment descriptors, etc.) needed to Deploy and run the system. The MDA also addresses the problem of middleware [10]. Middleware is a general term for any software that serves to glue together or to mediate between two separate and existing programs. Often found in a distributed environment, middleware is a layer above the operating system, above the application programming interface (API of Windows) but below the application program. Middleware masks some heterogeneity that programmers of distributed systems must deal with. Examples of famous middleware are CORBA, Enterprise Java Beans, XML/SOAP, COM+, and .NET. The goal of the MDA [10] is to separate business and application logic from its underlying execution platform technology so that changes in the underlying platform do not affect existing applications. The evolution from one model to another normally needs a model transformation that we explain in the next section two ways for transform from one model to another. The benefit of this approach [10] is that it raises the level of abstraction in software development at least in the early stages. Instead of going directly to platform-specific code, software developers focus on developing models that are specific to the application domain but independ-}

ent of the implementation platform. MDA, contrary to its name, are only a conceptual framework. It does not define any particular software architecture or any architectural style. And also If development [15] time should be longer for a first application or for a company without any development assets, development time should decrease for subsequent applications developed inside known domains and platforms.

V. Transformation in MDA

There are two schools in how transformation from one model to another can be achieved. These two schools are “elaborations” and “translations”. In the elaborations approach [14] is shown in Figure 2. The definition of the application is built up gradually as you progress through from PIM to PSM to Code. Once the PIM has been created, the tool generates a skeleton or first-cut PSM which the developer can then “elaborate” by adding further information or detail. Similarly, the tool generates the final code from the PSM, and this may also be elaborated. The elaborations approach [14] To MDA probably represents the mainstream. It is Essentially a mechanization of the familiar OO development approach, where first an analysis model (OOA) is produced, followed by design model (OOD) and finally code. In the traditional approach, however, there was no automation of the production of models, either forwards or backwards. MDA brings this automation. In the translations [15] as shown in the
The PIM is translated directly into the final code of the system by code generation. The transformation (or translation) of the PIM into the final code is performed by a sophisticated code generator, sometimes called a “Model Compiler”, and symbolized by the large arrow. It is driven by Generation Rules that describe how the elements of the PIM are to be represented in the final code. The PSM is an intermediate stage in the generation and is internal to the code generator. It is generally not visible or editable by the developer. A feature of the translations approach is that the downstream artifacts (PSM and Code) are not further elaborated or amended by hand. The PIM (plus the Generation Rules) are the full source of the generated system and there is no need to look at or amend the generated artifacts, in the same Way that there is no need to look at or amend the output from a programming language compiler. The translations approach derives, in the main, from work on real-time and embedded systems only. Finally, CIM for Telecom-Egypt training sector has considered several users including The PIM and other students, instructor, lab, materials, system administrator. CIM also shows the e-LMS system architecture and standard specification to be employed. UML will be used in this layer to model user requirements and their boundaries. PIM, in this model will give that is being expected to be provided by e-LMS database through design class diagram can be used to select the open source LMS platform. The model of that platform will enable implementation of a system with desired qualities and will promote the nature of mapping towards PSM, which is in this case a XML document platform. PSM, The e-LMS is a web based platform in order to carter for wide access to a big number of students in Telecom-Egypt that will be need to train. The functionality specified in the PIM is realized in a PSM through the application of some transformation. Modeling applications is our target in this case UML. We can relate the phases in OOSA&D with the conceptual framework created by OMG in MDA and by using UML diagrams as shown in figure 4.

Case Study : Modeling the system using MDA Approach

We use MDA Model to Model transformation model as described in fig4 as our framework in modeling our system “e-LMS for Telecom-Egypt training sector” using Uml tools and OOA&D

A. Background Study

First, we will explain how the traditional training systems work in Telecom-Egypt and then describe statement problems in this traditional system. Telecom-Egypt’s training sector consists of four general departments (as described in fig 5):

1. General dept. for planning & program design, where the training Materials and courses are designed.
2. General dept. for train Executions, responsible for assigning the suitable instructor to each course.
3. General dept. Evaluations & Exams, evaluate the training process which contains materials, instructors, labs, courses for all aspects.
4. General Dept. for Regional Institutes which controls and manages all institutes over Telecom- Egypt company in different cities and is responsible for labs where course are executed and support services needed for required training equipment.

in this research we will focus on The general department of evaluations and exams that evaluate all thing in Telecom-Egypt training sector - students, instructors, courses' materials and labs -. We try to model the optimal solutions to reach the highest per-
formance Using MDA Model. The training sector process begins by the training sector’s requests to other sectors for its courses that are needed to train their employees and collects those requests from other sectors then transforms it to courses as shown in step 1 in figure 6. This allows to translate to step 2 which is the execution process that includes assigning instructors, allocating the institute where the course will be executed and determining the schedule of the course. After executing and finishing the course we translate to step 3 - the last stage-in which the institute management distributes the feedback forms to students and instructors. When they finish filling the forms, the institute management collects it from them and sends them by fax to the main branch in Cairo. This feedback forms evaluate all training resources such as instructors, Labs, Materials and institute management. This is in brief how the Telecom-Egypt training sector works, but this traditional system has many problems that we will describe it in the next section.

B. Statement of Problems

1. The student doesn’t trust evaluator who works by default in the same institute where the course is executed.
2. The large amount of evaluation form papers are sent by fax to the main branch in Cairo. (This problem represents type of balance wastage for Telecom-Egypt training sector).
3. The management can’t get some important information from its training process such as de-
terminating the best instructor, best institute management......etc.

4. The train may be take week or many weeks and the shortage of specialist and engineer make it impossible to leave their work and go to training sector to take courses.

So in our research we will solve these problems by modeling the open e-LMS systems that solve these problems by using Unified Modeling Language (UML) and MDA based on Object Oriented Analysis and Design (OOAD).

C. MDA –First Step

The MDA’s First step that describe in fig 4 is computation independent model (CIM). A CIM [17] is a model of a system that shows the system in the environment in which it will operate, and thus it helps in presenting exactly what the system is expected to do. It is useful, as an aid to understanding a problem. In an MDA specification of a system CIM requirements should be traceable to the PIM and PSM constructs that implement them. A CIM might consist of requirements analysis and UML diagram called Use case diagram that we describe in fig 4.

➢ Requirement Analysis

A requirement is singular need detailing what product or service should be or do or which is what customer really wants [6]. we use different means of data collection that have been included interview with telecom-Egypt’s holders (administrators, students and instructor), physical observations, existing Documents to reach exactly what telecom-Egypt’s stake holders requirements need from e-LMS and we have concluded from our research these requirements:-

1. The course’s schedule sent from the main branch to the allocated institute where the course will be executed.

2. The charged person login the system using his or her account and create the course.

3. And then manage the course by enrolling students, instructors and allocating the lab into the course.

4. After enrolling students, instructors the system sends the course’s information to students and instructors automatically by emails.

5. The students login the system and evaluate the course after fishing it. These evaluation forms are stored into the database of this course.

6. The instructors login also the system and evaluate the course and put the exam’s result of the course.

7. General dept. For Evaluations &Exams in the main branch login the system and print the Summary reports for all courses in all institutes about the results and evaluation of the course.

After we determine our system’s requirements from interviews with management and ensure from them that requirements they already needed to solve the problem. We will go to build the use case diagram that match these requirements

➢ Use Case Diagram

Use Case Diagram: is a way to capture system functionality and requirement in UML. Use case diagram consist of name pieces of functionality, the person invoking the functionality which called as actor and possibly the element responsible for implementing the use case. Generally, use cases express the behavior of a system, model the functionality requirement of system using actors and use cases without the excessive detail which often confuses people with a less technical background [5]. A use case illustrate the overall function in the system and how the system should be respond in condition to a request of the user requirement [7]. Human interact with our system in different roles as: student, instructor and institute management, Assessment management, Exams management, trainers management. And we define each actor and describe what it means in our system:-

1. Institute management: as shown in fig 11, is the actor number one in our system, who starts the event by create course in his or her institute, and manage it by enroll students and instructors and he or she can also create account for students.

2. Instructor: as shown in fig 12 is the second actor in our system, who will teach the course and he
starts his role by login the system after institute management enrolls him or her into the course, he or she do evaluation of the course and put the exam’s results, view his or her historical course and he or she changes his profile.

3. Student: as shown in fig 12 is the actor number three in our system, who is a trainee of the course, he start his or her role by login the system after institute management enrolls him or her to system, he or she do evaluations of the course, view his or her historical courses and change also his profile.

4. Assessment management: This person starts his role after the course finishing and he view the courses’ evaluation in all institute and print the summary report which indicate the evaluate of instructor, lab, course’s materials, staff and he or she can also change his profile as shown in fig 13.

5. Exam management: This person starts his role after the course finishing and he view the courses exams’ result from all institute and print the summary report which indicate the final exam’s result and he or she can also change his profile.

6. System administrator: This person responsible for environmental system especially in handling database system. System administrator will generate the new variable and can change the value of variable in update database function. Generally, system maintenance can achieve all function in use case such as login / logout update form, view Telecom-Egypt training Sector data and manage it as shown in fig 14.

D. MDA –Second Step

The MDA’s Next step that describe in fig 4 is platform independent model (PIM). PIMs [17] represent the functionality and behavior of a system and capture only the application logic. A platform independent model is a view of a system from the platform independent viewpoint. A PIM exhibits a specified degree of platform independence so as to be suitable for use with a number of different platforms of similar type. A very common technique for achieving platform independence is to target a system model for a technology-neutral virtual machine [17]. A virtual machine or Middleware is defined as a set of parts and services (communications, scheduling, naming, etc.), which are defined Independently of any specific platform and which are realized in platform-specific ways on different platforms. Virtual machine is a platform, and such a model is Specific to that platform. But that model is platform independent with respect to the class of different platforms on which that virtual machine has been implemented. In fig 7 to 9 Imagine a class with a property “name”. To generate a PSM, one must decide to which platform it will be bound. If the choice is made to implement this in C#, the PSM would have to incorporate another private field (private string fName for example) and a public property Name.

![Fig.7. PIM and different PSM in different platforms](image)

```csharp
private string fName;
public string Name {
    get {
        return fName;
    }
    set {
        fName = value;
    }
}
```

![Fig.8. Platform specific implementation for C#](image)

```csharp
private String name;
public String getName() {
    return fName;
}

public void setName(String value) {
    fName = value;
}
```

![Fig.9. Platform specific implementation for java](image)

```java
private String name;
public String getName() {
    return fName;
}

public void setName(String value) {
    fName = value;
}
```

The examples show that, although the PIM and PSM are related, they are not the same. PIM is related to system analysis and system design stage [16]. UML will model e-LMS at different viewpoint levels of abstractions from the platform independent viewpoint. This model will give what is being expected to be provided by e-LMS to Telecom-Egypt Training sector. At this stage, the structure of the e-LMS through design class diagram can be used to select the open source LMS platform. A PIM might represent by UML class diagram show in fig 4, it shows the structure and behavior of program system. In class diagram contents using design elements such as classes, packages and objects, name, signature, and
properties. There are several types of relationship in class diagrams such as association, dependency, generalization, aggregation, composition and realization. Each relationship is represented in the diagram by a difference type of arrow [5]. In our system we use the use cases in the previous section to generate the class diagram but this transformation will be manual transformation. As shown in fig 15 Class diagram represent the structure and global of program, showing classes, interfaces, and their relationships. They help software developer by abstracting implementation details and presenting view of the program lines of code. Class diagram would help software maintainers to understand programs architecture and to locate places requiring modifications during maintenance [17]. In figure 15 we present our class diagram in entire system.

E. MDA -Third Step

The MDA’s third step that shown in fig 4 is platform specific model (PSM). UML class diagram can be constructed to represent the elements, relationships, and constraints of an Extensible Markup Language XML application visually and then transformed into one or more XML schemas, which serve as their PSM Counterparts shown in fig 4.

➢ XML

Extensible Markup Language is a meta-markup language made up of a set of tags to define and describe the contextual meaning of data [18]. XML overcomes some of the drawbacks of the more popular markup language, HTML (Hypertext Markup Language) by allowing the user to create and define his own tags, by giving him the flexibility to precisely code and search documents. The syntax of XML used in this paper is the one recommended by the World Wide Web consortium (W3C). XML uses tags to define different elements of a document and is case-sensitive. XML tags [19] can be classified into elements and attributes. An element is the main building block of XML and basically has name and content enclosed by a pair of tags. An element can have an attribute, which is a name-value pair attached to an element’s start tag. An element can contain other elements, attributes, and data types, and based on the items they carry, they form different content models. A simple or complex type defines the complete structure of elements and their counter-parts. A document in XML is a tree of elements, which can be parsed with an appropriate tool. An XML document looks like a hierarchical representation of data. This representation lends itself to be easily modeled using UML class diagrams. Similar to the rules of a diagram in UML, XML has Document Type Definitions (DTD), which are collections of rules describing elements and other markup objects [18]. A DTD defines the document structure with a list of legal elements. XML supports two classes of documents, well-formed and valid. The former conforms to XML syntax but does not have a DTD and can be checked only for syntax and not structure. The latter is a document that has a DTD and can be checked for both syntax and structure. More recently, XML schemas define the structure of an XML document [20]. Schemas have several advantages over DTDs, some of which are that, they are easier to read, and they provide ways to specify patterns for data, which cannot be done with a DTD. They are also extensible and can be reused in other schema definitions. XML schemas are written in XML so the need for using another language is eliminated. They also support. They also support data types, which make it easier to describe data and namespaces, which make it easier to manage extensibility. XML schema language is also called XML schema definition (XSD). A parser can use an XSD schema to verify if the document is valid.

➢ Transformation UML To XML

We present the results of our literature and tools survey here and also discuss the transformation from UML diagrams to XML. Carlson [3] discusses the transformation of an UML model to XML schema and the reverse engineering process based on XMI (XML Metadata Interchange) rules. XML is a specification Proposed by the OMG to exchange data over the Internet in a Standardized way and ensure consistency and compatibility between applications created in various environments. It combines the benefits of XML with object-oriented UML. The mapping itself is based on a pre-defined UML profile that adds XML-specific definitions to a UML model. Routledge et.al [10] discusses the mapping between UML class diagrams and XML schemas using the traditional three-level database design approach (conceptual, logical and physical design levels). At the conceptual level, UML class diagrams describe the objects and relationships. The logical-level is a direct one-to-one representation of the XML schema data structures in terms of a UML profile [21]. The UML profile tailors the language to specific XML schema concepts that carry specific semantic information, beyond what is usually expressed in traditional UML elements. The UML profile uses stereotypes, tagged values, and constraints for most of the XML schema elements that are mapped in the physical level. The physical level corresponds directly to the logical level representing the XML schema. The conceptual level UML class diagrams [22] are transformed successively into XML schemas. Specific XML schema concepts that carry specific semantic information, beyond what is usually
expressed in traditional UML elements. The UML profile uses stereotypes, tagged values, and constraints for most of the XML schema elements that are mapped in the physical level. The physical level corresponds directly to the logical level representing the XML schema. The conceptual level UML class diagrams are transformed successively into XML Schemas addresses the mapping of UML package diagrams [21]. Finally our survey conclude we can transform PIM model to PSM using XML language. There are many commercially available UML CASE tool converter were developed with standard UML such as, Rational Rose, Poseidon, and ArgoUML. We can use one of them to convert our e-LMS shown in fig 10.

Fig. 10. Mapping PIM to PSM

The e-LMS is a web based platform in order to cater for wide access to a big number of employees in different sector of Telecom-Egypt Training Sector. The functionality specified in the PIM is realized in a PSM through the application of some transformation. Web services applications is our target in this case which will run in an application server. Web services application is the nature of transformation from PIM to PSM. The transformation from PIM into PSM will be to express the UML into XML Metadata Interchange (XMI) using standard definitions expressed as XML Schema Definitions (XSD) as shown in fig. 12. XMI is an application of Extensible Markup Language (XML), which lends itself to transporting information that is highly internally referential. XML is applied to transport UML models by generating a special XSD through applying the rules of XMI to the concrete UML metamodel. The general mechanism applied within OMG to transport meta-information is XMI [16]. Finally, XML document that produced by using any UML 2 XML converter as previously mentioned in “Transformation UML to XML section” will be mapped to application using any programming languages interfaces, code and Structured Query Language (SQL queries).

IV. Conclusion and Future Work

Software design approach is applied simply to produce a software solution to a problem. To produce quality software we need to through understanding the require-
ments that satisfy the user’s needs. MDA approach offers an improvement in understanding the problems not only at development time but also later in the development process, in particular during maintenance. By using as many available tools, such as UML and XML. We were able to show, that a tool allowing Transformations from abstract models of system to concrete implementation of e-LMS of telecom-Egypt training sector is possible. Although the solution presented in the form of the MDA is not fully implemented, it will be developed and researched further. The proposed solution presented in this paper is to model the e-LMS in an abstract manner. Through the help of the MDA Support for Telecom-Egypt training sector the paper shows that it is possible to derive an implementation of e-LMS from a PIM model. Because it is not the intention of this paper to further complicate the development of e-LMS but rather to model them. Other benefits that we conclude from our research, Because MDA development focuses primarily on the functionality and behavior of complex applications, the analyst, when designing the PIM, does not need to shift his attention to idiosyncrasies of the technology platform on which the application will be implemented. This in fact, is the task of the developer that has a very good understating and overall knowledge of the platform he is implementing with after the creation of models and study of the architecture, the software analyst can make modifications related to any part of the application. If the application was already written and a crucial flaw would appear (e.g. a performance issue in error handling) it could take a while to redefine the architecture and rewrite the impacted classes, in turn wasting effort of both the architect and the developer. Using the MDA however, the analyst could simply modify the PIM and regenerate the application with no other effort from the developers.

An Object-Oriented system development approach is adopted in this research. The object-oriented analysis (OOA) technique used is based on the Model Driven Architecture (MDA) using Unified Modeling Language (UML) techniques for the design. Several types of diagrams such as use-case, design class diagrams will be employed. For a web-based system, an extensible markup language (XML) technology is applied, which maps with the UML to create a dynamic, interactive web application. Modeling will be obtained by mapping UML and XML schema. A high-level design process will follow that evolves into a more detailed design i.e. implementation and testing stage. This will involve code generation using high-level language. Code from XML document will be automatically generated.

References

Fig. 11. Course management
Fig. 12. Course evaluation

Fig. 13. Evaluation and Exams management
Fig. 14. Registration Systems Management
Fig. 15. Telecom-Egypt Training Sector class Diagram