The Impact of the Computational Independent Model for Enterprise Information System Development

Yashwant Singh
Jaypee University of IT, Waknaghat,
Himachal Pradesh, INDIA

Dr. Manu Sood
Himachal Pradesh University, Shimla,
Himachal Pradesh, INDIA

ABSTRACT
Models are of main concern in Model Driven Architecture (MDA). MDA models are precise and written in well defined language. The aim of defining these consistent, standardized and high level models is transformation from Computational Independent model to Platform Independent Model, Platform Independent Model to Platform Specific Model, and eventually to executable code. Researchers have proposed various approaches for transforming Platform Independent Model to Platform Specific Models in MDA with detailed explanations of both input and output models, but artifacts of Computational Independent Model have been ignored in the MDA approach of software development by current researchers. This paper is an attempt to highlight the impacts of Computational Independent Model in the process of software development under MDA software development strategy. It provides insights into the specification of the business processes, stakeholders and dependencies between processes with the help of a suitable real life example. The business system which is independent of software technology is described by various structured and dynamic UML diagrams.

General Terms
Model, Platform, Enterprise Information System, Requirements, Use Case Diagram, Activity Diagram and Sequence Diagram.

Keywords
Model Driven Architecture, Transformation, Computational Independent model, Platform Independent Model, Platform Specific Model.

1. INTRODUCTION
The process of software development of Enterprise Information Systems are improving day by day, but the improvement process is slow because current strategies of developing an enterprise information systems are code centric, thus are less flexible, more complex, very costly and have long life cycles. A software developer should prefers to spend more time in building models rather than writing codes, because intelligently modeled enterprise system reduces the development time, cost and complexity [1].

In order to shift from code centric software development approach to model centric software development approach a new framework owned by Object Management Group (OMG) [2] namely Model Driven Architecture (MDA) has been proposed. MDA is a software development strategy which focuses on Models, Transformations and code generation. MDA defines an approach of specification of enterprise information system that segregates functionality of system specification from functionality of system implementation for a specific technology platform. MDA gives complete freedom to enterprise application developer to develop a system based on business logic and data rather than focusing on middleware, hardware and operating system. Figure 1 illustrates the basic concepts of various models in MDA.

Various factors such as software as a service, service oriented architecture and value chain oriented business is shifting the software development process toward MDA. This paper is an attempt to explore Computational Independent Model in detail due to its need as input to Platform Independent Model in development of software using MDA strategy of software development, and CIM is designed by using various UML diagram. It also provides an overview of, development process based on MDA. A real life example has been worked out to illustrate CIM and its importance.

CIM captures the requirements of the Enterprise System being computerized. All the requirements including business rules, functional, non functional, user and system are modeled under this CIM without emphasizing on any particular technology. The CIM consider only what are the business/user needs of enterprise system not how enterprise system will be designed and in which platform will it be implemented. The importance of CIM is that it is input to Platform Independent Model which is more technical than Business Model and used as an input to the Transformation Tool for transforming Platform Independent Model into Platform Specific Model.

The remainder of this paper is organized as follows: section 2 provides a brief overview of Model driven Architecture with its models and process of transformation. Section 3 presents CIM and its importance in development of enterprise system using MDA strategy. The concepts of CIM with the help of an example have been illustrated in section 4. Continuing the illustration section 5 presents the Platform Independent Model which takes CIM as input and section 6 presents conclusion.

2. MODEL DRIVEN ARCHITECTURE AND ITS MODELS
In the last decade complexity and risk of software development have increased tremendously. Modeling has made great impact to deal with complex and risk oriented software development. Models are used for analysis (analogues to CIM of MDA), design (analogous to PIM of MDA) and implementation (analogous to PSM of MDA) of software artifacts in software life cycle. The figure 1 shows the architecture of various models of MDA and transformation among them. Many researchers have proposed various approaches for transforming Platform Independent Model

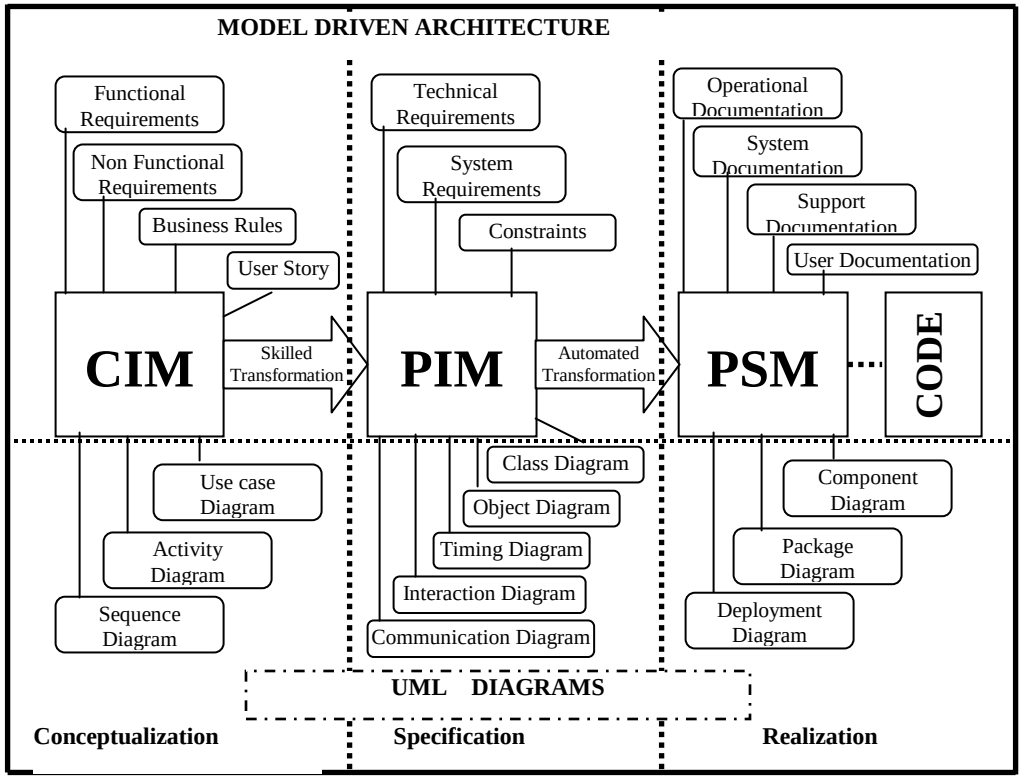

Figure 1. Concepts of Model Driven Architecture.

to Platform Specific Model in MDA but artifacts of CIM have been somehow ignored by current development in the MDA approach of software development [3]. It is pertinent to note that to meet the business and user requirements, the importance of Computational Independent Model and transition from Computational independent model to Platform Independent Model is critical. Figure 2 shows the three layer structure of CIM which depicts the artifacts of CIM on the basis of problem domain, application architecture and organizational characteristics. The enterprise application functionalities not only depend on business environment and user requirements but also on organizational characteristics.

2.1 MDA Software life Cycle

The MDA software development life cycle is a five-step process:

1. Create a CIM; CIM of a system describes domain and requirements of the system with the help of various UML diagrams. For capturing the exact requirements of the enterprise system, the Use case diagram, activity diagram, and sequence diagram are to be drawn [4].
2. Develop a PIM; the PIM does not include any details of particular platform; it captures the domain's key concepts technically. A software developer can represent these concepts through models in different modeling languages like UML which includes class diagram, object diagram, timing diagram, interaction diagram and communication diagram. PIM consists of a general representation to capture the semantics of many
3. different domains, which should also be precise enough to support transformation into target PSM [5].
4. Transformation of PIM into Relational PSM, EJB PSM and Web PSM, by using the rules of transformation of PIM to PSM [6].
5.
6. Combining the different PSM's and generate the code from PSM; and
7. Deploy the enterprise system on to a specific environment.

3. COMPUTATIONAL INDEPENDENT MODEL

The MDA based software development strategy derives its requirements from business organizations. In this strategy, the use and exchange of information is driven by business needs more than a software solution. MDA clearly presents a business motivated approach to the development of software system. This software development approach starts with the first model of MDA namely Computational Independent Model (CIM) which describes business ambience and business requirements. CIM is then transformed to the next model named as Platform Independent Model (PIM) which explicitly explains services and interfaces provided by software system without considering any technology platform. The PIM is further transformed into Platform Specific Model (PSM) for realization of software system to specific technology [7].

The CIM explains through its various diagrams about the software system that is where it will get operated and what this system will achieve.

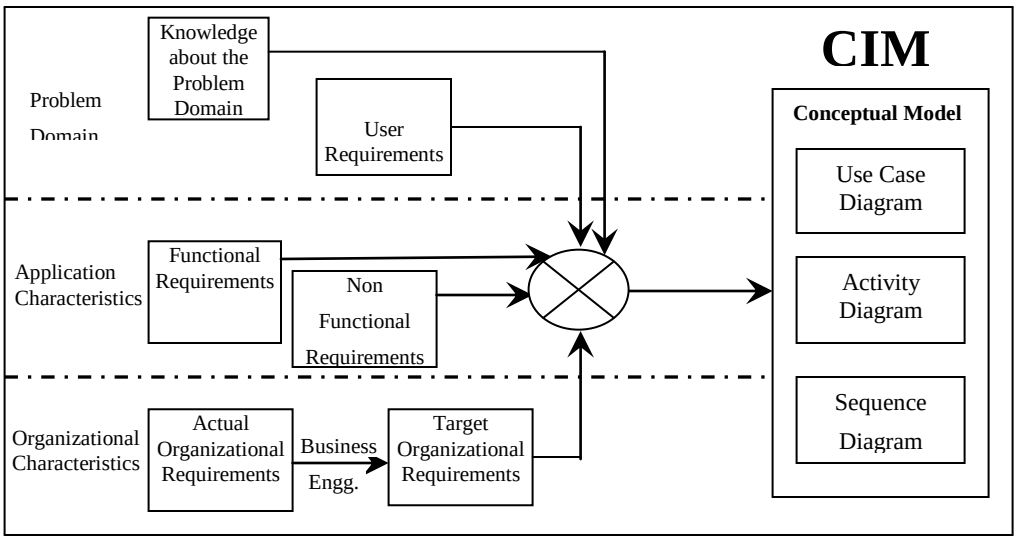

Figure 2. Computational Independent Model Concepts.

It is useful to understand various requirements of the enterprise systems. In traditional software development system requirement engineering is divided into two parts [8]: 1) Requirement Development which consists of elicitation, analysis, specification and verification of requirements and 2) Requirement Management which consists of configuration management and monitoring of requirement implementation. Generally the software system requirements are categorized as;

- User Requirements
- Organizational Requirements
- Functional Requirements
- Non Functional Requirements

All these requirements in CIM of MDA approach can be depicted by three diagrams namely Use Case Diagram, Activity Diagram and Sequence Diagram.

3.1 Impact of CIM in MDA

The various factors which lead the software projects to failure are modeled under Computational Independent Model are following:

- Incorrect interpretation of requirements; which cause wrong allocation of resources and additional cost.
- Fluctuating requirements; requirements are unstable, additional requirements are added during the different phases.
- Doubt and uncertainty regarding certain requirements.
- An imperfect requirement consisting of data and reports incomplete information about the system.
- Contradictory objectives by different software developers.

The problem domain requirement shown in the figure 2 is the first level which analyzes the requirements in widest range of any application. A solution of any problem can be developed with systematic way by specifying problem domain with full parameters and mapping. The second layer shown in figure 2 is application characteristics which consist of functional and non functional requirements. Minimal complexity, reusability, portability, loose coupling, use of standard techniques and ease of maintenance are the principles of good application characteristics. The third layer in figure 2 depicts the organizational characteristics which illustrates common objective, coordination of work, risk-taking within limits, accountability and hierarchy.

These three layers are interlinked and can be combined into UML diagrams. Use Case Diagram describes the interaction between user and system, Activity Diagram represents workflow and actions and Sequence diagram shows how system processes operates one another. Combination of all these diagrams makes Computational independent Model which is the input model of Platform Independent Model.

The concepts of a CIM are shown in figure 1. and include three types of activities:

1. Eliciting and/or gathering the requirements from the users,
2. Analysis of the requirements which consists of checking for any ambiguity, incompleteness and uncertainty, and
3. Documentation of the requirements in the form of Use case Diagram, Activity Diagram and Sequence Diagram.

The following are the various types of requirements obtained under Computational Independent Model:

3.1.1 User Requirements

The software system deliverables specified by the user without considering any implementation platform are called user requirements or business requirements. These are modeled in CIM with the help of Use Case Diagrams. Good specifications of user's requirements are needed for successful software project completion. The correct specifications of business requirements in CIM in terms of what system should do, is done by the business users rather than computer system developers who may not have an exact idea of how various business activities are performed.

3.1.2 Functional Requirements

The specification of the desired behavior of an Enterprise System in CIM in terms of business processes, functionality and implementation is called functional requirements. These requirements represent the functionalities of the system under consideration that are needed to fulfill the objectives of the system.

3.1.3 Non Functional Requirements

The specifications of measurable aspects of the system in CIM are called non functional requirements. These measurable aspects include reliability, availability, efficiency, reusability, portability, scalability and flexibility etc.

3.1.4 Organizational Requirements

The specification of the organizational hierarchy of Enterprise System in CIM, which consists of structure, processes and classes of the users of the system.

4. EXAMPLE: BUSENESS MODELLING

The illustration of the Computational Independent Model in MDA software development life cycle process can be explained with the help of an example of Physician's Activity System (PAS) [3]. The PAS is a enterprise software used to manage various activities related to the check up of patients, the treatment of patients, medical tests of the patients and management of various financial transactions involved.

To capture the requirements of this example PAS in a model corresponding to CIM, MDA software developer must understand first business organization which consists of actual organizational requirements and target organizational requirements and then functional and non functional requirements and further user and system requirements.. Since a CIM represents the domain without considering any particular system implementation or technology, it would remain the same even if the system were implemented manually, rather than automated [8,9,10]. After interviewing the

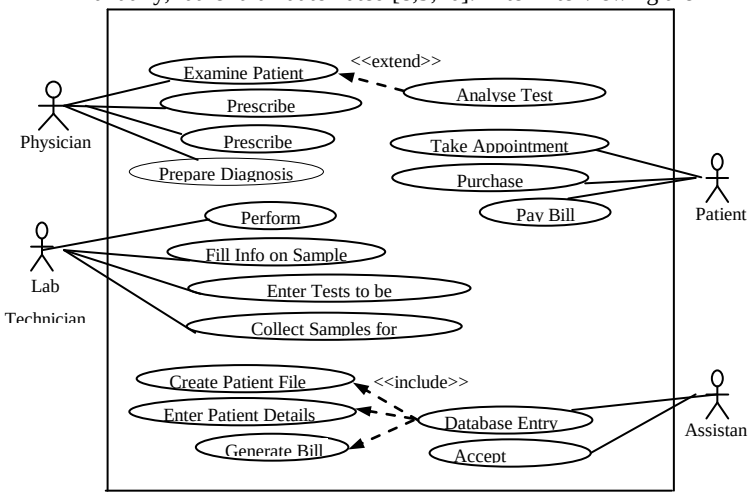

Figure 3. Use case diagram of PAS (part of CIM).

stakeholders of PAS and understanding its business needs, an MDA software developer specifies the following activities of its different actors that may occur in the due course of working of PAS and may used while modeling CIM with its different diagrams like use case diagram, activity diagram and sequence diagrams:

1. The assistant gives the appointment time of Physician to patients according to available time, he also creates a new patient file (for new patients) or opens up an already existing file (for patients already registered with PAS) to note the detailed information of the incoming patient and sends the patient to doctor for initial check up.

2. The Physician or doctor prepares the prescription mentioning tests and medicines, if any, and; procedure of treatment .If test is required Physician sends the Patients to Lab. Technician.

3. The lab technician performs the prescribed tests on the patient and after recording the information of all these tests sends the reports to the doctor again.

4. Doctor examines the findings of all the tests if any and prescribes the treatment to the patient.

6. The assistant generates the bill for the patient that includes registration fees, and tests and medicine charges, if any.

7. The patient clears the bill by making the payment and walks away.

The different actors or users of the system and their activities, characteristics are as follows [3]:

1. Physician is a person who examines the patient, prescribes test and medicine for patients; and prepares the diagnosis.

2. Lab technician is a person who is responsible for: a) performing medical tests on patients, and b) preparing the tests reports.

3. Assistant is a person who is responsible for: a) entering the details of patient, b) generating bills for the consultation fee, test charges and medicine charges, and c) Accepting payments from patients, d) Allocation of appointment time.

4. Patient is a person who visits the doctor for the medical treatment of an ailment.

As part of CIM, figure 2 shows activity diagram of PAS and figure 3 shows the UML Use Case diagram of PAS

Computational Independent Model is a model which is used as the input to the Platform Independent Model therefore it must be properly designed. The CIM designed by Use case Diagram, Activity Diagram and Sequence Diagram model the all structured and dynamic aspects of the system. The Use case diagram in figure 3 represents the requirements from the user's point of view. Also representation of use cases helps the system analyst the behaviors of the system and visualization of use case by UML can get additional information from users. The UML Activity Diagram shown in figure 4 is similar to flowchart of programming languages. The activity diagram shows the sequence of activities, action points and various divisions of actions. It shows the business operations and business process activities and becomes useful of system analyst to make analysis. The structured model of the software system can be represented by use case diagram and activity diagram. The figures 5 to 7 shows sequence diagram which provides the dynamic view of the software systems consists of objects, time lines and messages between time lines with direction.

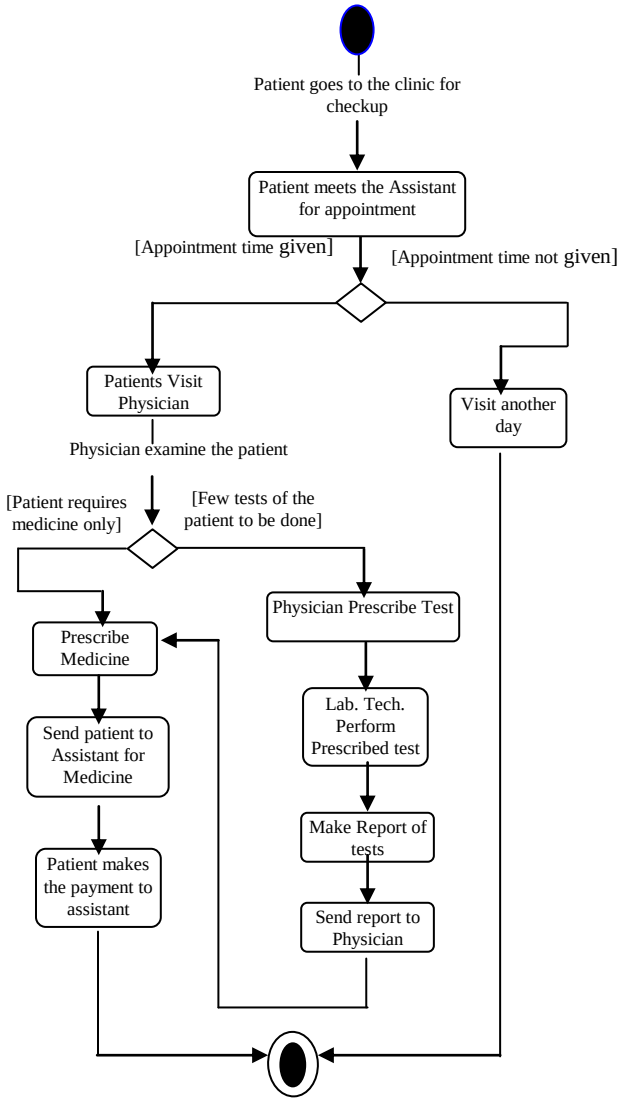

Figure 4. Activity Diagram of PAS (part of CIM).

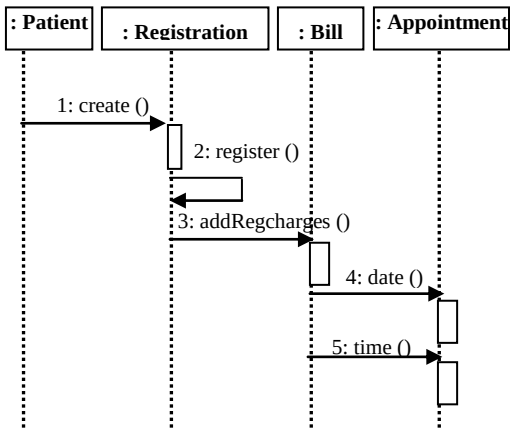

Figure5. Sequence Diagram for Appointment of PAS (part of CIM)

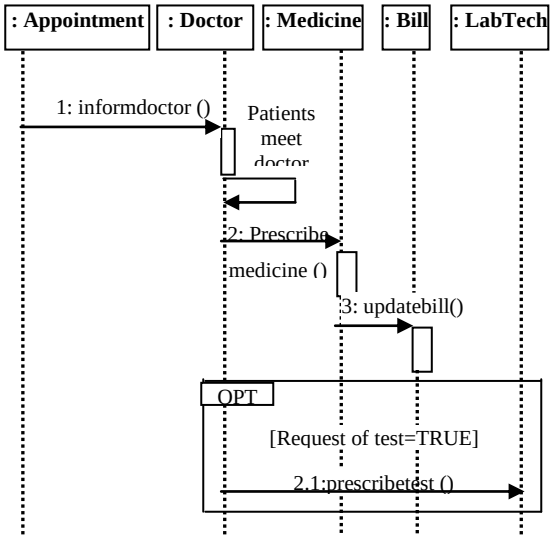

Figure6. Sequence Diagram for Prescription of PAS (part of CIM)

5. PLATFORM INDEPENDENT MODEL

The PIM does not include any technology specific details; it obtains the domain's key concepts from Computational Independent Model. The main goal of the PIM is to create the abstract concept of the system using the vocabulary of the domain captured in different UML diagrams created under CIM. A software developer can represent this concept through UML models in general. PIM demands a general representation to capture the semantics of many different domains because it is to be transformed further, it should also be precise enough to support transformation into Platform Specific model. As an OMG standard, MDA uses the UML models as its core representation. IBM's Rational Rose 2003, Visual paradigm and OMG's XML Metadata Interchange (XMI) are thus popular tools that support the process [14]. Figure 7 shows the UML class diagram which is independent of any platform and derived from CIM.

The key features of PIM those are necessary to transform the PIM into PSM are as follows [2,4,5,15]:

- Independent of implementation platform/technology
- Formation of abstract model.
- Business requirements are specified using UML diagrams.
- System is modeled from the viewpoint of how it best supports the end user requirements.
- Describes system behavior, functional and nonfunctional aspects independent of the computing environment & implementation technologies. Can be reused across multiple platforms.

6. CONCLUSION

Enterprise Information System is the main artifact of the enterprise information construction. The study of how to improve the quality, longevity, cost of production and success rate of information system is absolutely necessary and significant. The MDA software development approach in the recent years has been able to improve all these factors by separating the concern through abstractions at various levels. This paper has shown that the notion of various models, transformation among models and software development life cycle process of MDA. The source

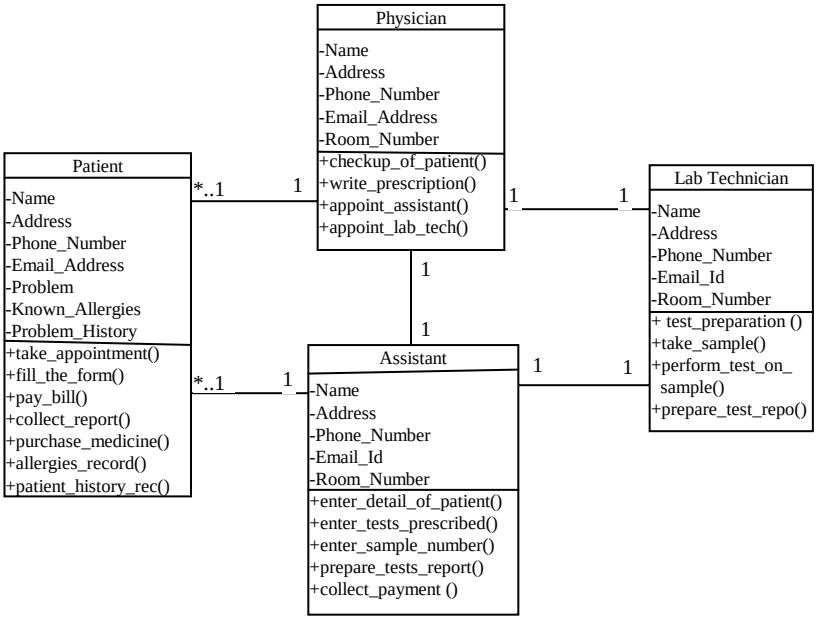

Figure 7. Platform Independent Model

model on the transformation from analysis to design namely Computational Independent Model (CIM) in MDA which covers all types of requirements for the enterprise system has been presented in detail with the help of various UML diagrams. To get the best transformation results in the design process of software, the quality requirements with perception of real world (Modeled in CIM) is very important. It has been showed that CIM which consists of use case diagram, activity diagram and sequence diagram informs the software developer about structured and dynamic aspects of the system but does not compel the design decisions in a specific technology way.

7. REFERENCES

[1] D. S. Frankel, The Model Driven Architecture: Applying MDA to Enterprise Computing, OMG Press, 2003.

[2] "Object Management Group (OMG): Model Driven Architecture (MDA)" www.omg.com/mda, Sep 16, 2008.

[3] Y. Singh and M. Sood, "Model Driven Architecture: A Perspective", IEEE International Advance Computing Conference (IACC'08), Thapar University Patiala, India, pp,1644-1652, 6-7th March 2008.

[4] S. S. Alhir, "Understanding the Model Driven Architecture (MDA)", http://www.methodsandtools.com/archive/archive.php?id=5, 2003.

[5] G. A. Lewis and L. Wrage, "Model Problems in Technologies for Interoperability: Model Driven Architecture," Software Engineering Institute, Carnegie Mellon University Pittsburgh, PA, Technical Note CMU/SEI-2005-TN-022, May 2005.

[6] T. O. Meservy and K. D. Fenstermacher, "Transforming Software Development: An MDA Road Map," IEEE Computer, vol. 38 no. 9, pp.52-58, 2005.

[7] Sami Beydeda, Matthias Book, Volker Gruhn, "Model-Driven Software Development",ISBN-10 3- 540-25613-X Springer Berlin Heidelberg New York, 2005.

[8] Y. CHE, G.WANG, X.X.WEN and B.Y.REN, "Research on Computational Independent Model in the Enterprise Information System Development Mode Based on Model Driven and Software Component", IEEE International Conference on Interoperability for Enterprise Software and Application, China, pp,85-89, 2009.

[9] Janis Osis, Erika Asnina and Andrejs Grave, " Proceedings of the 10th International Conference on Information System Implementation and Modeling" , Czech Republic, 2007.

[10] J. Mukerji and J. Miller, "MDA Guide Version 1.01", http://www.omg.org/docs/omg/03-06-01.pdf, Jun 12, 2003.

[11] T. Mens, K. Czarnecki and P.V. Gorp, "A Texonomy of Model Transformations," Proc. of International workshop on Graph and Model Transformation (GraMoT), 2005.

[12] A. Kleppe, J. Warmer and W. Bast, MDA Explained: The Model Driven Architecture, Practice and Promise, Pearson Education, Inc. , April 2003.

[13] Y. Liu and Y. Ma, "An Approach for MDA Model Transformation Based on JEE Platform", 4th WiCOM international conference, 12-14 Sep.,2008.

[14] R. Heckle and M. Lohman, "Towards Model Driven Testing," Electronic Notes in Theoretical Computer Science 82(6), 2003.

[15] DEMIR Ahmet, "Comparison of UML 2.0 and DSLs in the Context of MDD", Diploma Thesis, Technische University at Muchen, 2005.