International Journal of Database Theory and Application
Vol.7, No.4 (2014), pp. 155-168
http://dx.doi.org/10.14257/ijdta.2014.7.4.12

Modeling Business for MDA Software Paradigm

Deren Yang[1,2] , Min Liu[2] and Bo Li[2]

[1]*School of Science, Ningxia Medical University, Yinchuan, China*
[2]*School of Computer Science and Engineering, Beifang University of Nationalities, Yinchuan, China*
[1]*ydr@tom.com*

Abstract

Aimed at resolving the second software crisis due to increasingly technical complexities and diversities of software development methods, Model Driven Architecture (MDA) was proposed in the late 2000. However, because of its lacking an effective and well-known modeling process or method, it is difficult to apply and promote MDA in the industry. In this paper, MDA is extended with a business model, and corresponding theory and presentation of the business model are explored, especially the dependencies and transformation mechanism of its sub-models are discussed. And a case of shopping is shown. Based on the business model, CIM is traced. And its advantages are discussed by compared with the Rational Unified Process (RUP) and Dines' triptych paradigm. This research is helpful to apply MDA in the industry.

Keywords: *business model; MDA; forms & reports; business object; modeling mechanism*

1. Introduction

Proposed and standardized by the Object Management Group (OMG), MDA software paradigm is a software development framework designed to respond to the second software crisis, which is a result of diversified programming languages and more sophisticated technology platforms used. Originally, MDA software paradigm contains three models, *i.e.*, a computation independent model (CIM), a platform independent model (PIM) and a platform specific model (PSM). There are some relationships among models, such as inheritance, conversion, traceability and evolution [1-2]. The advantage of MDA is transforming software development into modeling by raising abstraction level, but it main disadvantage is vagueness among the models, for example, the definition and source of CIM is not clear [3], about which no consensus is reached [4]. In order to solve these problems, a business model (BM) for MDA is proposed and its model mechanism is explored.

This paper is organized as follows: some related researches are reviewed in Section 2. An early extended MDA software paradigm and its modeling mechanism are shown in Section 3. Business model and its three sub-models, namely business functions model, business process model and business object model, and its corresponding modeling mechanism are detailed in Section 4. A comparative study of the business model with RUP and Dines' triptych paradigm are discussed in Section 5. Finally, conclusions and author's future researches are arranged in Section 6.

ISSN: 2005-4270 IJDTA

2. A Brief Overview of Related Researches

Since OMG did not provide a software methodology or a software process for MDA software paradigm, its modeling mechanism and process have been a focus of research for years. OMG did recommend Unified Modeling Language (UML) as a modeling tool for MDA[5], but as to which UML diagrams are used to build the models of MDA, no consensus is reached [4], so it is difficult to apply and promote MDA software methodology in the industry [4,6]. Some issues are as follows.

Firstly, the semantics, representation and sources of CIM are ambiguous. Oasis considered CIM to include business model and requirement model [7], Laforcade took CIM as a business model [8], Kirikova thought CIM as an information processing model [4], Gherbi thought CIM as a requirement model, and PIM as analysis and design models [9], Luciana considered PIM as a business model [10], Nikiforova divided PIM into a business layer and an application layer [11]. In addition, models and corresponding representations are also inconsistent, as Yamin used class diagram to describe PIM [5], and Sharifi used activity diagram to model business [12].

Secondly, no consensus was reached on business model. Proposed by a branch of IBM, as an iterative software development process framework, Rational Unified Process (RUP) did have a business model, but it was only an option. The model is composed of two sub-models, of which one is a business use case model describing external interaction with the organization, and the other is business analysis model also known as business object model [12]. The advantage of RUP is that the business model was proposed, while the disadvantage is that the business model is incomplete and lacking a sub-model for business process. Dines proposed a triptych paradigm including domain engineering, requirements engineering and software design. And a business process, which contains entities, functions, events and behaviors, is included in the domain engineering [13]. Dines did propose a business process, but it was just a concept only. Author had already mentioned a business model with its components and its importance [4], but did not study it details.

So, in order to trace CIM and treat above issues, business model and corresponding modeling mechanism should be explored.

3. MDA and It's an Early Extension

3.1. MDA and Corresponding Models

As a new software development paradigm and a set of standards and technologies [6], MDA is based on a core principle known as "everything is a model"; and as a conceptual framework its ultimate aim is the automatic conversion from models to codes; and its essence is to create executable codes based on software architecture models. Models are conducive to the stakeholders to understand, design and realize software.

MDA simplifies design by ignoring unrelated details, and raise the abstraction level in software design, and through separation of architecture to achieve portability, interoperability and reusability. MDA software paradigm contains three models, *i.e.*, a computation independent model (CIM), a platform independent model (PIM) and a platform specific model (PSM).

3.2. A Modeling Mechanism for the Extended MDA

As described before, the only three models are not enough for MDA. Author had proposed an implementation model (IM) in implementation domain to extend MDA aimed to improve MDA modeling process [6]. Its main idea is described below.

3.2.1. The Models and Corresponding Domains: In MDA, a model corresponds to a domain. A domain refers to the scope of a problem or task. A domain should be described with some basic elements and the fundamental relations among the elements. Through the separation of concerns on domains, MDA builds different models in different domains. In MDA, there are three domains, *i.e.*, a problem domain, a solution domain and an implementation domain. A problem domain comes from a business domain. The relationship between the models and domains are: CIM corresponds to the problem domain, PIM corresponds to the analysis sub-domain of the solution domain, PSM corresponds to the design sub-domains of the solution domain, and IM corresponds to the implementation domain [6].

3.2.2. The Models and Corresponding Stages: In MDA, different models are built at different stages of modeling process as shown in Table 1, and the relationship between model and stage are as follows:

① CIM describes software requirements at the requirement stage, and mainly focuses on software function requirements.

② PIM is created at the software analysis stage, which explores software internal details, but has nothing to do with specific platform.

③ PSM is established at detailed design stage, which describes model according to the specific platform.

④ IM is a model built at the implementation stage, including code and data.

3.2.3. The Models and Corresponding Representations: The implementation of MDA needs a software methodology and a modeling language to support [6]. The object-oriented approach and UML may well be a good choice. The steps of modeling are as follows:

① CIM is built with use case diagram, and is further explored with use case specification to prescribe GUI and necessary scenes.

② Based on CIM, at the analysis stage PIM is built with robustness diagram which is an unessential diagram in UML.

③ Based on PIM, during the design stage, PSM is built with class diagram, sequence diagram and database table schema.

④ Based on PSM, at implementation stage, IM is built with a specific programming language, such as Java or C #, and data.

By using UML, the modeling process is optimized, for example, in order to bridge the design gap between the CIM and PSM, a robustness diagram is used to establish PIM [14].

In modeling process, the result of the immediately previous stage is the exact basis of the latter's, which embodies the relations between models, such as inheritance and convertibility. It is conducive to reuse models. The models, stages, domains and languages are shown in Table 1 [15].

Table 1. The Models, Stages, Domains and Languages

Model	Stage	Domain		UML diagram
CIM	Requirement stage	Problem domain		Use case diagram, Use case specification
PIM	Analysis stage	Solution domain	Analysis	Robust diagram
PSM	Detailed design stage		Design	Class diagram, Sequence diagrams, Table schema
IM	Implementation stage	Implementation domain		Java or C #

3.3. A Newly Extended MDA Software Paradigm

As shown from Table 1, CIM did specify software requirements, but it's unclear whether CIM belongs to business domain or software requirements and what its source is.

3.3.1 A Proposed Business Model for the Extended MDA: A business is a service provided by an agency or an organization to its customers. Software is designed for auto realizing the real-world business, so the source of CIM is a business model in a business domain.

3.3.2. The Relations among Models and Corresponding Dependence: Similar to relations shown in Table 1, a business model is added as a source of CIM, The relations among models, domains, stages and languages are shown in Table 2 with extra information for every model on basic elements in corresponding domain and UML diagrams.

Table 2. The Relations among Models, Domains, Stages and Languages

Model	Stage	Domain	UML diagram	Basic elements
BM	Business requirements	Business domain	Use case diagram, Activity diagram, Class diagram	Business actors, Business use case, Actions, Objects
CIM	Software requirement	Requirement domain	Use case diagram, Use case specification	System use case, Actors, GUI, paths
PIM	System analysis	Analysis domain	Robustness diagram	Boundary Object, Control Objects, Entity Object
PSM	Detail Design	Design domain	Class diagram, Sequence diagrams, Table schema	Actors, Objects, Messages, Table schema
IM	Implementation	Implementation domain	Java or C #	Class file, SQL Statement

3.3.3. The Transformation Mechanism among Models: by Adding BM to MDA, not only optimizes MDA modeling process, but also makes CIM traceable. Based on UML diagram, the conversion mechanism among models are as follows:

① BM is a description of business, which is a preliminary analysis of the implementation of business, and is built with UML use case diagram, activity diagram and class diagram only with name.

② Based on BM, CIM prescribes software requirements, and is built with use case diagram which naturally evolved from business activity diagram, and use case specification.

③ PIM explores three kinds of object prototypes, and is built with robustness diagram which is based on event flow in use case specification and further optimizes the former.

④ Based on PIM, PSM concerns the detailed design of software, and is built with class diagrams, sequence diagrams and database table schema. It is noticed that sequence diagram comes from robustness diagram, and operations are assigned to object here, class diagram evolves from business object diagram and is completed with above method.

⑤ Based on PSM, IM concerns about specific implementing platform, such as programming languages, and is built with code and data.

4. The Business Model

Based on the author's early research, business model consists of three sub-models which are business function model, business process models and business object model [4]. Business function model is used to capture service provided by an organization to its customers, and business process model is used to design use case scenario, and business object model is used to describe business objects used or produced in the scenario, and relations among them.

A useful mechanism for modeling business is vital, so is the conversion among the models. The mechanism is as follows: firstly, defining business function model with use case diagram; secondly, analyzing business processes with activity diagrams for each business use case diagram; finally, extracting object, business forms and reports from business process model and further extracting business objects from the forms and reports to generate business object model.

It is worthy to note that extracting business objects from the forms and reports should be based on the low representational gap (LRG) principle in order to take business objects as similar to prototypes for software objects as possible.

The mechanism for modeling business is shown as Figure 1.

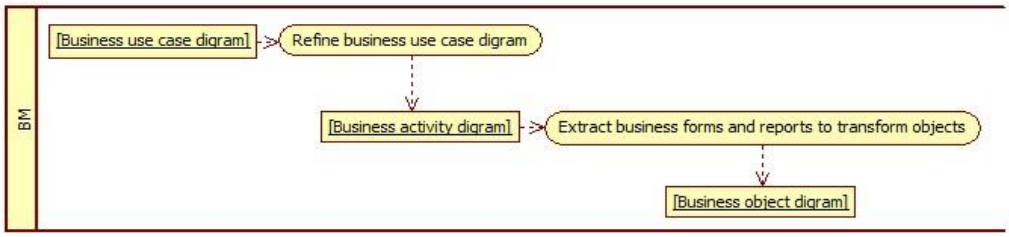

Figure 1. A Mechanism for Modeling Business

4.1. The Purposes of Business Model

Business modeling is a high level of abstraction beyond software, and aims to define and represent a social system, based on which software can be designed. Specifically, business modeling can achieve two kinds of purposes, *i.e.*, the purposes for business and the purposes for building IT systems.

The purposes for business are as follows:

① The business modeling describes how to develop a vision of a new organization, and based on this vision, business use case model and business object model are explored to define process, role and responsibility of organizations;

② The key elements are deeply understood with dynamic and structure in the existing business, the problem domain and its potential are identified as the basis of improving structure and operations in existing business;

③ Showing innovative business structure, experimenting with new business ideas, and learning the concept of a competitive company used.

The purposes for building IT systems are as follows:

① Determining the scope of the software system as early as possible to prevent unnecessary or unrelated work, and to ensure that customers, end users and developers agree on with the business goals of an organization, and to reach a consensus on implementation process among stakeholders;

② Understanding the structure and mechanism of the target organization for identifying the improvement of potential, and the organization's structure and dynamic;

③ As the basis for supporting software development, exporting and modeling software requirements for organizations in order to identify outsourcing opportunities.

4.2. The Business Function Model

4.2.1 Its theory: Business function model is the highest abstraction of business. It captures services provided by an organization to its customers, and describes what should an organization should do for its customers in order to realize its goals. By building the model, the initial scope of business will be defined clearly.

4.2.2 Its presentation: Business function model is built with UML use case diagram, known as business use case diagram in the context.

Use case can also be used to express the interaction of user and software, known as system use case diagram in the context. In order to distinguish them, the former is called system use case, and the latter is known as business use case.

A business use case, also known as an essential use case, aims at make use cases simple and abstract, in order to capture the intents of its customers.

The main elements of the business use case diagram are business use case and business actor, which are used to express the interaction of customers and business organizations.

The role of business use case diagram is to express customer’s demands on business, which is the basis of the follow-up modeling. In addition, it makes organizations’ objectives clear.

4.3. The Business Process Model

4.3.1 Its Theory: The business process model describes a series of actions and flows fluently performed by business workers to serve its customers (business actor). It’s a detailed description of business in order to make stakeholders have a clear understanding of business process. By building the model, the business process can be optimized, and the resources required and results produced in the process will be clearly shown.

4.3.2. Its Presentation: Business process model is represented with UML activity diagram [16].

Every business use case diagram can be converted to one or more business activity diagrams according to its granularity.

A business activity diagram consists of nodes and flows.

There are three kinds of nodes, i.e. action nodes, control nodes and object nodes; and the corresponding roles are as follows:

① The action node is a smallest unit which is indivisible. They receive process data and output data to the next nodes, and the data including some business forms and reports.

② The control nodes have the control effect, which include decision-making in the plurality of parallel flows.

③ Object nodes are tokens which hold data temporarily, and they'll wait for a suitable opportunity to circulation in an activity diagram.

Flows are divided into two types, which are control flows and object flows, and the roles of the flows are as follows:

① The boundary of the control flows connect with activities, which are used to describe the action that should not be terminated before they reach the target action. Only control nodes can pass through control flows.

② The boundary of the object flows connect object nodes and input actions. Only objects and data can pass through the object flows.

The business process model focuses on inner actions, flows, resources, artifacts and results in the process, of which, part or whole can be implemented with software, in another word, software requirements can be extracted from the process model. So a business process model can be further converted to system use case diagram in CIM.

4.4. The Business Object Model

4.4.1. Its Theory: The business object model, also known as domain model, is an abstract model in the context of business domain, which aims at capturing the most important entity object types. Actually, the business objects, derived from forms and reports in corresponding business activity diagram, are different from pure objects in RUP. The forms and reports are something like users' view in a Database, which are in form of 1NF in database theory, and will be taken as a basis for abstracting object entities in PIM in order to form robustness diagram which will be refined to sequence diagram to establish PSM later.

4.4.2. Its Presentation: Business object model is built with UML class diagram, in which objects only have corresponding names because of lacking necessary information about attributes and methods. In fact, it is the prototype of software class diagram which includes attributes and operations in PSM, and also the source of software database table schema.

The mainly elements of the business object diagram are object names and relations.

① The object names are actors, or the objects which are extracted from forms and reports in a business process model.

② The relations among objects are only the simplest association, which will be explored in PIM and detailed in PSM.

4.5. A Case Study: Shopping

Based on a traditional case such as shopping, the sub-models of the business model are detailed as follows.

4.5.1. Business Function Model: The system to be modeled is based on business in an organization or its sub-units. Business function model is a model of the first abstract level, which is a conceptual model of what business could do. In business function model, actors are outsider of the organization, but they are different, so should be distinguished. For example,

the business actors are customers, suppliers and other organizational units, while, internal employees are located within the boundaries of the organization, therefore they cannot be defined as business actors. It is noticed that each business use case represents a business process, and business actors are the people who are outside organizations business, and they directly start the business. Business actors should have one at least, and business events should also have one at least.

In the case of shopping, business actors are organizational customers, and the corresponding business process is shopping. Business use case diagram is shown in Figure 2.

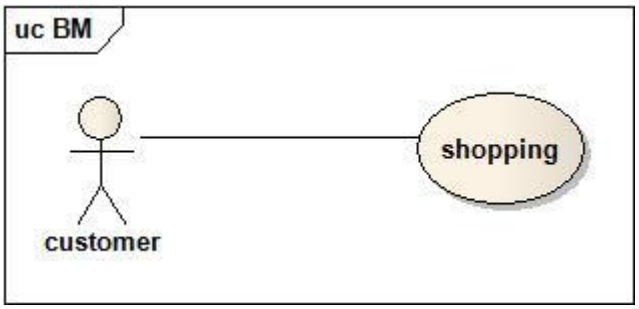

Figure 2. The Business Use Case Diagram

4.5.2. Business Process Model: In the case of "shopping", the customer needs pre-sales consulting, sale and post-sales service as shown in Figure 3.

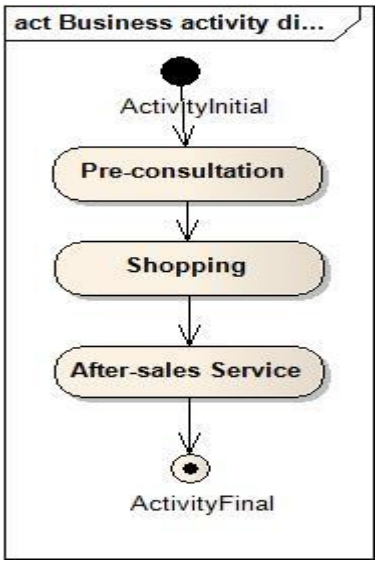

Figure 3. The Business Activity Diagram

Figure 3 is a highly abstract business activity diagram, and the three activities can be appropriately decomposed into corresponding sub-activities represented with detailed activity diagrams.

In the pre-sales consulting, involves two actors, *i.e.*, a customer and a service, are involved, as shown in different lanes. The specific process is shown as Figure 4.

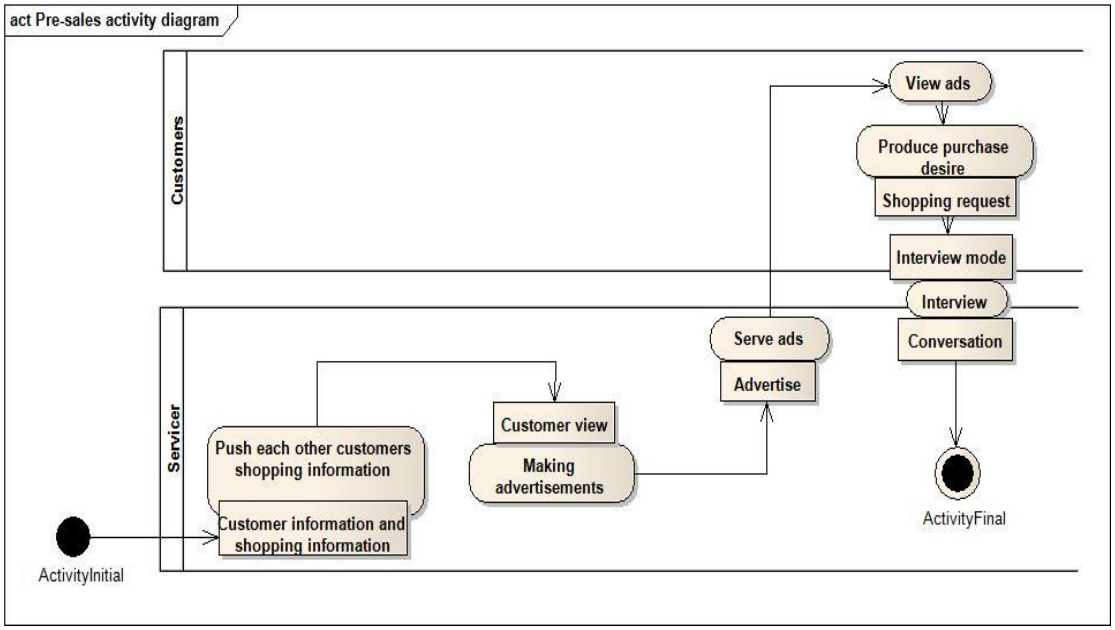

Figure 4. The Pre-sales Activity Diagram

The same as the pre-sales consulting, the sale activity diagram is shown as Figure 5.

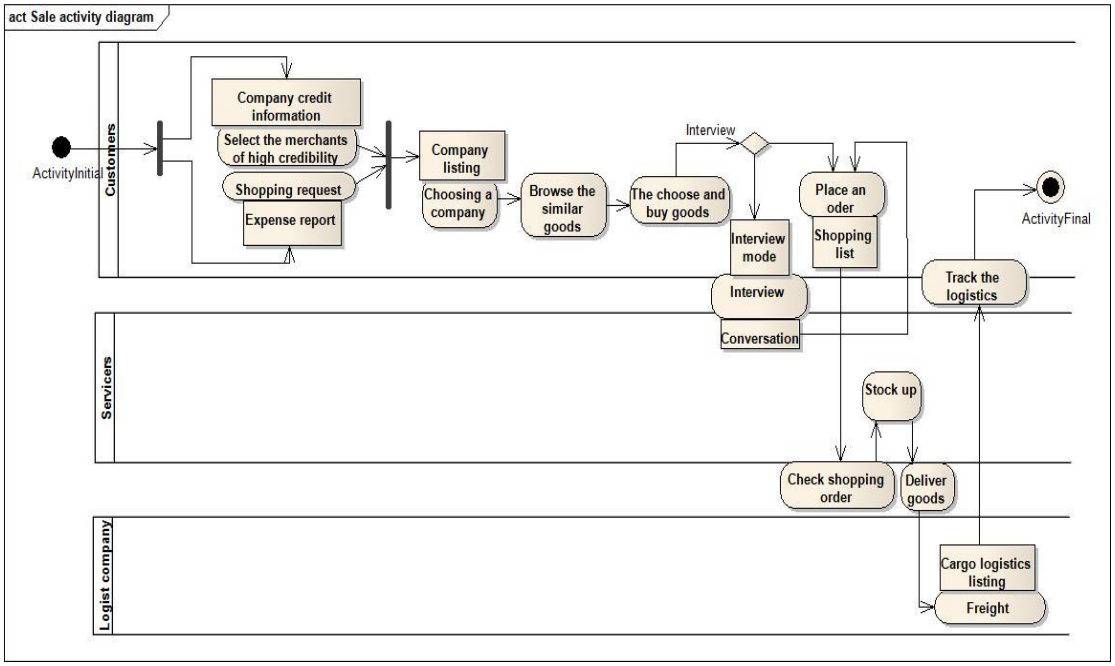

Figure 5. The Sales Activity Diagram

Similar to two previous activity diagrams, the post-sale process activity diagram is shown as Figure 6.

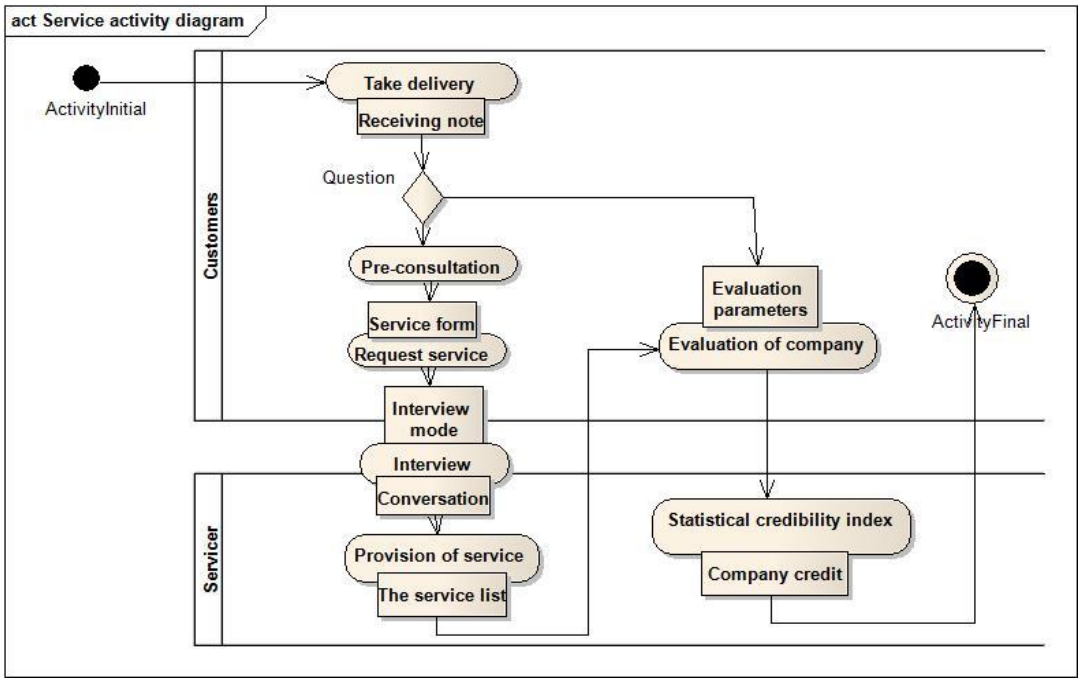

Figure 6. The Service Activity Diagram

4.5.3. Business Object Model: From object flows in above business activity diagrams, extracting no business objects which are very original, neither based on 3NF in the database tables or pure objects. In fact, they are some business related forms and/or reports. In the case of shopping, the so called object nodes in business process model are extracted and further are converted into pure objects, which are main components in business object model, then based on the relations among the objects to build business object model.

Figure 7 is extracted from the business process model, which shows the relationships between so-called objects. This kind objects is not real objects, so it need to be further elaborated in order to get a real business object diagram, as shown in Figure 8. In fact, Figure 8 is the prototype of the class diagram in PSM.

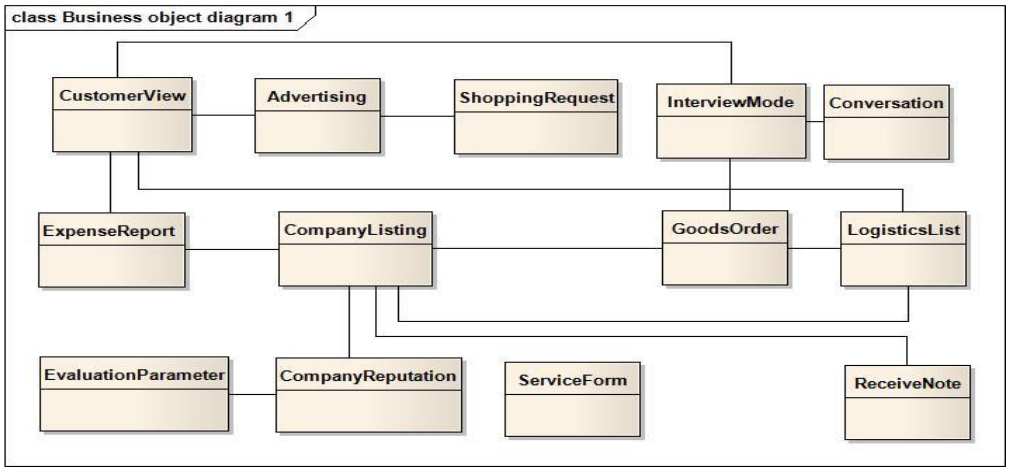

Figure 7. The Original Business Object Diagram

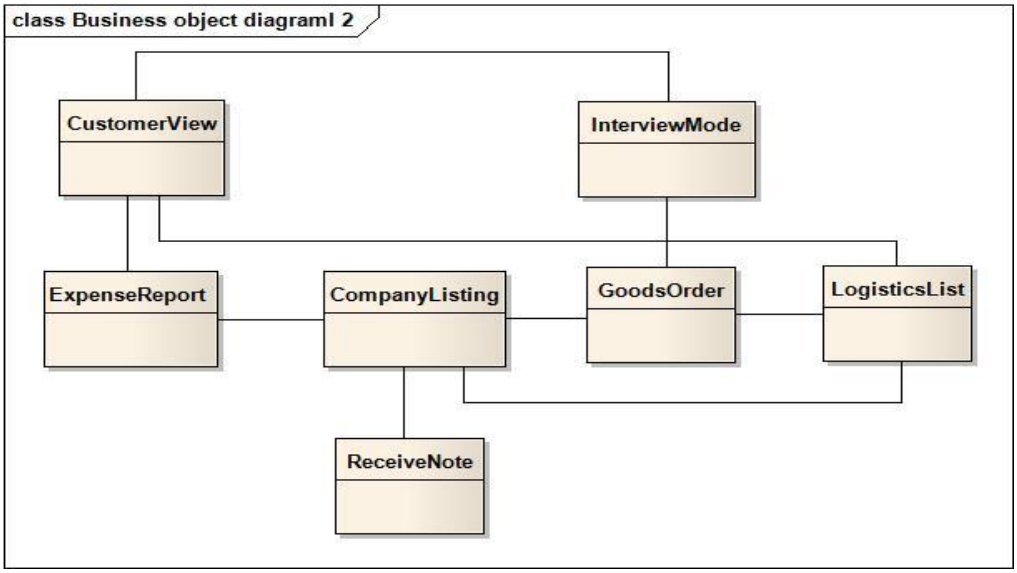

Figure 8. The Optimized Business Object Diagram

5. Discussion

5.1. The Source and Traceability of CIM

The five models of MDA are closely related. Among them, PSM is a detailed design model, and CIM aims at recording critical software requirements. If CIM is a business model, then PIM is a requirements model, so between PIM and PSM exists a design gap. Therefore, CIM should be a requirement model. In fact, software is designed and developed to service business in an organization, and theoretically CIM should correspondingly be based on a business model.

5.1.1. The Source of CIM: Any organization or company has its businesses. A business model is a description containing business function, process and some necessary artifacts. Business model is an indispensable prerequisite to build CIM, in other words, CIM comes from corresponding business model.

5.1.2. The Semantics and Composition CIM: CIM is a description for software requirements, which is unrelated to the internal details. CIM consists of system use case diagram and use case specification. The system use case diagram comes from actions in business activity diagram. The system use case specification is a document which includes general process, special process and exception process; meanwhile, it is the exact source of robustness diagram in PIM.

5.2. Compared with RUP and Dines'

5.2.1. Compared with RUP: In RUP, business model includes business use case model and business object model. In the business use case model, the relations between the business actors and business use case are described, and in business object model the objects and relations among them are shown, but the origin of the object is not detailed.

From the horizontal perspective, the objects in RUP business object diagram are pure objects, but in our method the original objects are business forms and reports which like user views not based on 3NF, which are much close to actual situations. From the vertical perspective, because of lacking business process model in RUP, in the development methodology exists a gap between business function model and business analysis model. Our business model and RUP' method are compared in Table 3.

Table 3. Our Method and RUP' Method

	Our proposal		RUP	
	Models	Representation	Models	Representation
Business Model	Function model	Use case diagrams	Use case model	Use case diagrams
	Process model	Activity diagram	Object model	Object diagram
	Object model	Object diagram		

5.2.2. Compared with the Method of Dines': Dines has pointed out a triptych paradigm, *i.e.*, domain engineering, requirements engineering and software design. Business process model, which contains entities, functions, events and behaviors, are included in domain engineering [13].

From horizontal aspects, a business process was proposed, which merged entities, functions, and behaviors into some documents. But its model is vague, and only represented in the form of some summaries or documents. From vertical aspects, Dines' domain engineering lacks levels, and the modeling mechanism is not perfect. Our business model and Dines' are compared in Table 4.

Table 4. Our Method and Dines'

	Our proposal		Dines'
	Models	Representation	In domain engineering Business process includes entities, functions, events and behaviors.
Business Model	Business function model	Business use case diagrams	
	Business process model	Business activity diagram	
	Business object model	Business object diagram	

5.3. The Advantages of our New Method

Based on the original MDA, a business model and its modeling mechanism are proposed, which effectively solve some related problems. Based on human thinking mode, business model contains business function model, business process model and business object model, which are built with UML use case diagram, activity diagram and class diagram respectively. The advantages of our method are as follows.

Firstly, the modeling mechanism of MDA is extended and optimized, which makes CIM traceable, meanwhile, eliminating the ambiguity among the original MDA models. Secondly, business modeling principles and representations are explored. Finally, the objects in business object diagram originally are user views and reports which need further to be discomposed in to real objects, which is corresponding with the object in software design.

6. Conclusion

Based on the MDA models, in the paper, a business model is taken as the source of CIM, and its modeling mechanism is explored in details. The three sub-models of business model, *i.e.*, business function model, business process model and business object model are built with UML diagrams. This extension is conducive to research CIM' traceability, and compared with the related researches, the method has obvious advantages and operability. More research is needed to explore the mechanism of automatic implementing MDA software paradigm in future.

Acknowledgements

The research is supported by Project of Chunhui Plan (Z2011052) and Special Talents Project of Ningxia Medical University (XT200913).

References

[1] N. Anquetil, U. Kulesza, R. Mitschke and A. Moreira, "A Model-Driven Traceability Framework for Software Product Lines", Software and Systems Modeling, vol. 9, **(2009**), pp. 427-451.
[2] D. R. Yang and M. Xue, "On Software Process Paradigm and Its Constraint Mechanism", Proceedings of the 2nd International Conference on Software Engineering and Service Sciences, Beijing, China, July, **(2011)**, pp. 15-17.
[3] M. Kirikova, A. Finke and J. Grundspenki, "What is CIM: an Information System Perspective", Advances in Databases and Information Systems, 5968, **(2010)**, pp. 169-176.
[4] D. R. Yang, S. G. Wang and M. Liu, "Models and Corresponding Dependences In MDA", Proceedings of the 9th International Conference on Fuzzy Systems and Knowledge Discovery, Sichuan, China, **(2012)** May, pp. 29-31.
[5] M. Yamin, V. Zuna and M. A. Bugami, "Requirements Analysis and Traceability at CIM Level", Software Engineering & Applications, vol. 3, **(2010)**, pp. 845-851.
[6] D. R. Yang, M. Liu and S. G. Wang, "Object-Oriented Methodology Meets MDA Software Paradigm", Proceedings of the 3rd International Conference on Software Engineering and Service Science, Beijing, China, **(2012)** June, pp. 22-24.
[7] E. Asnina, J. Osis, in Model-Driven Domain Analysis and Software Development: Architectures and Functions, Edited J. Osis and E. Asnina, IGI Global Publishers , **(2010)**, pp. 40-64.
[8] P. Laforcade, B. Zendagui and V. Barré, A Domain-Specific-Modeling Approach to Support Scenarios-Based Instructional Design, 5192 **(2008)**, pp. 185–196.
[9] T. Gherbi, I. Borne and D. Meslati, MDE and Mobile Agents: Another Reflection on Agent Migration, Proceedings of the 11th International Conference on Computer Modeling and Simulation, Cambridge, UK, March, **(2009**), pp**.**25-27.
[10] L. N. Leal, P. F. Pires, M. L. M. Campos and F. C. Delicato, Natural MDA: Controlled Natural Language for Action Specifications on Model Driven Development, 4275 **(2006)**, pp. 551–568.
[11] O. Nikiforova, Two Hemisphere Model Driven Approach for Generation of UML Class Diagram in the Context of MDA, e-Informatica Software Engineering Jouanal, vol. 3**, (2009)**, pp. 59-72.
[12] H. R. Sharifi and M. Mohsenzadeh, A New Method for Generating CIM Using Business and Requirement Models, World of Computer Science and Information Technology Journal, vol, 2, **(2012)**, pp. 8-12.
[13] D. Bjørner, Editor, Software Engineering 3: Domains, Requirements, and Software Design, Springer (2006).
[14] D. R. Yang, F. L. Su and T. Zhou, Applying Robustness Analysis to MDA Software Process Paradigm. International Symposium on Instrumentation & Measurement, Sensor Network and Automation, Sanya, China, August 25-28,**(2012)**,pp. 419-422.
[15] D. R. Yang, M. Liu and Z. H. Gu, Modeling Principles of the Sequence Diagram and Its Application in MDA Software Paradigm. Advanced Engineering Forum, vol. 6, iss. 7, **(2012)**, pp. 15-19.
[16] Object Management Group: OMG Unified Modeling Language (OMG UML), Superstructure -Version 2.4.1.Available at: http ://www.om g.org/spec/UML /2.4.1/Superstructure/PDF/ (2011)11. Accessed on **1-11-2013**.

Authors

Deren Yang, Ph.D, he is a professor of Ningxia Medical University. He is a Master's tutor of Ningxia Medical University and Beifang University of Nationalities. His main research fields include Software Engineering, System Analysis and Integration.

Min Liu, he is a Master student, School of Computer Science and Engineering, Beifang University of Nationalities. Her main research field is Information System Analysis and Integration.

Bo Li, he is a Master student, School of Computer Science and Engineering , Beifang University of Nationalities. His main research field is Information System Analysis and Integration.