IJAMML 3:1 (2015) 51-67 September 2015
ISSN: 2394-2258
Available at http://scientificadvances.co.in
DOI: http://dx.doi.org/10.18642/ijamml_7100121539

PLATFORM-INDEPENDENT MODELING RECONFIGURABLE ENTERPRISE SOFTWARE AND APPLICATION

Hai-Bo Li and Jia Li

College of Computer Science & Technology, Huaqiao University, Xiamen, 361021,
P. R. China

Abstract

Nowadays, there are many factors to influence competitive advantage for enterprises, including increased market dynamics, shorter product lifecycles, and a higher customer involvement. Enterprise software and applications (ESA), as a way of IT (Information Technology), should have a higher agility to keep up with these changes, so as to produce significant influence to the competition. However, the traditional software development has no enough capacity to meet the requirement. Model-driven architecture (MDA) could shift the focus of iterative development to a higher level of abstraction. Of all the levels of MDA, the platform-independent model is the most important part because it lies in the middle of MDA and can dialogue with the models of both its higher level and lower level. Further, the platform-independent model can be used to generate codes. Inspired by MDA, a platform-independent modeling for ESA, called ICE-PIM (Interoperable Configurable Executable Platform-Independent Model), is proposed. Firstly, by analyzing the characteristic of ESA, a model of business object (BO) is proposed, which is composed of five diagrams, BO integrated diagram, BO data diagram, BO class diagram, and BO state diagram. Next, based on BO, ICEWf (Interoperable Configurable Executable Workflow), as a method for modeling business process is proposed. There are nine kinds of elements in ICEWf by

[*] Corresponding author.

E-mail address: lihaibo@hqu.edu.cn (Hai-Bo Li).

2010 Mathematics Subject Classification: 68M14, 68Q85, 68U35.
Submitted by Jianqiang Gao.
Received September 8, 2015

extracting process-related elements. Then, an interoperability of ICEWf with UML modeling is discussed, which is an actual standard for modeling language. The family of models can be transformed to executable code and workflow conveniently, so that the requirement for rapid changes of ESA can be satisfied very well. Finally, a case study is discussed to show the capacity of modeling business.

Keywords: platform-independent modeling, enterprise software and applications, business object, workflow, business process.

1. Introduction

In fierce market competition environment, to improve competitive advantage for enterprises, enterprise interoperability and collaboration make enterprise management and business processes grow more complex. Information technology is one of the important approaches to realize the management and improve business processes. In such dynamic environment, enterprise software and applications (ESA) should have enough flexibility to satisfy the requirement. However, ESA must undergo a long-term and iterative development in traditional software development mode. Such ESA can not satisfy the requirement any more.

In order to improve the agility of ESA, the focus of iterative development has been shifted to a higher level of abstraction by the approach of model-driven architecture (MDA) [1, 2]. The MDA is a new way of writing specifications, based on a platform-independent model. The MDA focuses primarily on the functionality and behaviour of a distributed application or system, not the technology in which it will be implemented [3, 4]. UML model can be considered as a definitive platform-independent base to realize the specification [5]. However, from the perspective of mapping to code, it has many inadequacies in enterprise modeling. UML is not agile enough to follow the changes of business requirement. It is strongly aimed at an object-oriented vision of the world. It is good at computing and not at describing the business of an enterprise. As a key enabling technology for the MDA, a new method for enterprise modeling is required urgently.

The object-oriented approach seems advantageous to extend its use to upstream activities, so as to attain a uniform overall approach to analysis, design, and implementation [6]. Several object-oriented methods provide some guidelines for moving from analysis to design [7, 8], especially the original work by Coad and Yourdon [9] and some work on the use of UML [10]. Coad and Yourdon view design as adding implementation details to the model developed at the analysis stage. These details include human interaction, task management (e.g., multitasking requirements), and data management. Denis and Wixom [10] suggest how to specify system requirements with use case diagrams in order to drive the development process. For each use case, a class diagram is employed to specify object classes and their relationships, state charts are employed to specify object states, sequence or collaboration diagram are utilized to specify object interactions, and activity diagrams are used to specify flow of activities. However, both methods suffer from two main weaknesses. First, the models constructed in the analysis phase already include information system constructs, thus they do not represent the business independent of the information system solution. Second, they do not specify the process of deriving the information system model from the earlier enterprise model.

Some modeling methods focus on attributes of objects. In [11], an attribute-based, object-oriented approach for performing the change impact analysis tasks is presented. The approach models the integrated content of a product by characterizing its components and associated requirements with attributes and linkages. Some modeling methods focus on event of objects. In [12], a method is proposed to make extensive use of language constructs, such as constraints, derived types, derivation rules, type specializations and operations, which are present in all complete object-oriented conceptual modeling languages. Some modeling methods focus on features. In [13], a feature-oriented software development framework is presented.

Therefore, until now enterprise modeling methods still have many inadequacies. Although some object-oriented methods provide some generalized rules on how to identify potential object classes, they usually do not provide specific guidelines on identifying specific business object (BO) of enterprises, as well as its properties, states, activities, and communications. Identifying business object arbitrarily causes an increasing difficulty of system analysis and design.

Workflow is a perspective of business process. A workflow model based on BO should integrate all activities in a temporal order by the characteristics of BO. WfMC (Workflow Management Coalition) [14] has promulgated the conceptions, terms and production criterions. Another technology to solve business change is software component. Nowadays, successful application pattern for software component, such as CORBA, EJB, and DCOM are developed [15]. However, what we need today is a more scalable and configurable ESA.

In this paper, we propose an approach for enterprise modeling, called ICE-PIM (Interoperable Configurable Executable Platform-Independent Model), which including a BO modeling and a workflow modeling. To achieve the goal, we first analyze the characteristics of ESA, and then propose the concept of business object and the identification method. Next, we propose a method of workflow modeling, called ICEWf. Finally, we model and discuss a case.

The rest of the paper is organized as follows. In Section 2, characteristics of object-oriented business are analyzed. In Sections 3, the modeling rules of BO and ICEWf are presented. A practical case study is discussed to show our experience in modeling in Section 4. In Section 5, the interoperability of workflow model is discussed. Finally, in Section 6, we present a summary.

2. Object-Oriented Characteristic of ESA

Almost all ESAs, such as ERP and SCM system, focus on handling various informations of management and manufacturing, which are abstracted from various forms or reports. Moreover, the design of ESAs always start with analyzing forms and reports, including data structures of the forms, dependecies between the forms, operations on the forms, and external association between forms. For example, there are many attributes in a form of purchase requirement, e.g., serial number, requriement department, tabulator, date, bill of material and their requirement date, etc. There are many activities when processing the form, e.g., creating a new form, auditing a form, and submitted a form. Every activity is executed via a serial of fine-grained operations, called business operation. For example, the operations, such as appending, modifying and deleting a record of a table in a database, etc. are composed when editing a purchase requirement form. Being handled by activities or operations, the state of the form will be changed. For example, the form is under the state “editing” after creating, under the state “audited” when auditing finished.

The objects which are abstracted from forms or reports are called business object (BO). The BOs are filled in, transmitted and processed daily. From user perspective, BO refers to an aggregation composed of various information entities, including items of forms or reports, activities, operations, and states. From ESA perspective, BO refers to an integrated object carrying unique identify, lifecycle, data sets, and operation sets.

Generally, a BO involves a core data set and its many secondary data sets. Each data set almost has an operation set accordingly which are used to do business by users. BO’s lifecycle is described by its internal or other BO’s states. An activity is a set of operations which act on a whole or a part of data items of a BO.

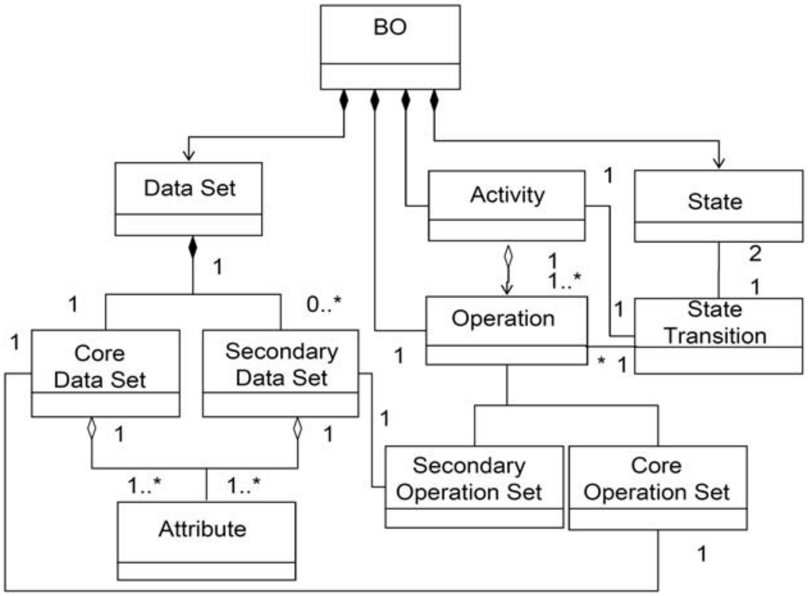


Figure 1. Meta model of business object.

Based on the analysis above, a meta-model of BO is proposed, as shown in Figure 1. BO can be defined as follows.

Definition 1 (Business object)**.** A business object is an abstraction of a form or report, denoted as *bo* = (*ID*, *DS*, *OP*, *S*, *Act*, *T*), where *ID* is a unique identifier of *bo*, *DS* is a set of data in *bo*, *DS* = (*CDS*, *ADS*), *CDS* and *ADS* are core data set and secondary data set, respectively, *OP* is a set of business operations in *bo*, composed of *Cop* and many other $AOp_i(i = 1, 2, \dots, n)$, which are associated with *CDS* and *ADS*, respectively, *S* is a set of states in *bo*, which are represented by attribute value(s), in which *S* = (*LS*, *AS*), *LS* is internal state set and *AS* is other secondary state set, *Act* is a set of business activities in *bo*, for any $act_i \in Act$, $act_i = \{op_1, \dots, op_m\}$ is a set of business operations, $T = \{s_i \to s_j | i = 1, \dots n,\ j = 1, \dots, n\}$ is a set of state transitions in *bo*.

We can conclude that a BO is: (1) an abstraction from forms or reports, (2) an encapsulation of core and secondary data sets, operation sets and state sets, (3) independent of other BOs when software developers design, and (4) independent of certain platforms under which ESAs can be coded by various program languages.

3. ESA Modeling – ICE-PIM

In this section, a platform-independent modeling ICE-PIM is discussed. ICE-PIM is composed two models mainly: BO model and workflow model. BO model is composed of three models: BO data diagram, BO class diagram, and BO state diagram.

3.1. BO data diagram

Each entity in a BO data diagram is described by entity name, entity ID, core attribute area, accessorial attribute area and association, as shown in Figure 2. Each entity in BO data diagram is linked to an attribute definition table which describes the map from attributes to database, including tables, field names, data types and other constraint conditions and compute formulas.

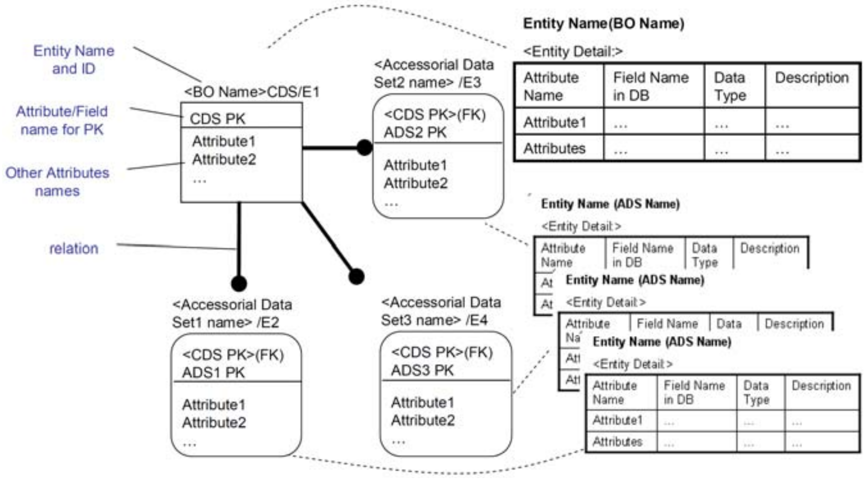
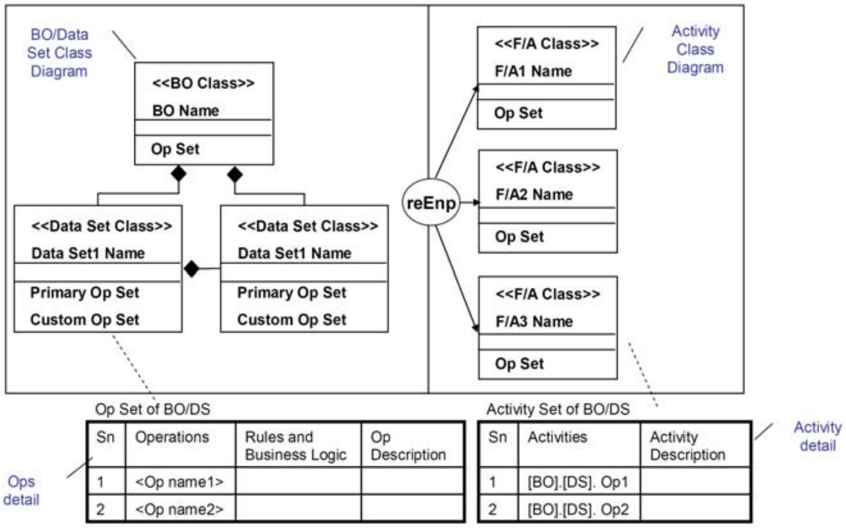


Figure 2. BO data diagram.

3.2. BO class diagram

BO class diagram is composed of three parts, as shown in Figure 3. The left part describes data set and operation set of a BO by UML class diagram, right part describes function sets and activity sets which encapsulate operation sets in UML class diagram. The middle part is the mapping relationship. The data set is composed of primary operation set and custom operation set. Each primary operation set and custom operation set are linked to a detail description, including the operation name, rules and business logic, activity, operation description, and activity description.



Op Set of BO/DS

Sn	Operations	Rules and Business Logic	Op Description
1	<Op name1>		
2	<Op name2>		

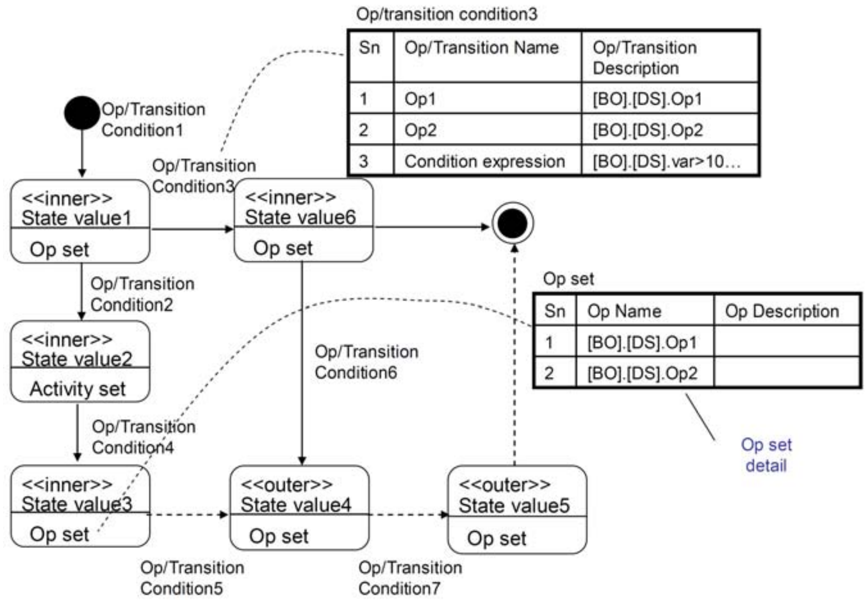
Activity Set of BO/DS

Sn	Activities	Activity Description
1	[BO].[DS]. Op1	
2	[BO].[DS]. Op2	

Figure 3. BO class diagram.

3.3. BO state diagram

BO state diagram describes states, operation under each state and transition between states. BO state diagram follows the criterion of UML state diagram, as shown in Figure 4. Each state transition is linked to detail description about operations and conditions. Each operation set is linked to a detail description about all operations. States can be classified into initial state, internal state, external state, and final state. The change of internal states is caused by the change of BO attributes. The change of external states is caused by the change of the other BOs.



Op/transition condition3

Sn	Op/Transition Name	Op/Transition Description
1	Op1	[BO].[DS].Op1
2	Op2	[BO].[DS].Op2
3	Condition expression	[BO].[DS].var>10…

Op set

Sn	Op Name	Op Description
1	[BO].[DS].Op1	
2	[BO].[DS].Op2	

Figure 4. BO state diagram.

3.4. ICEWf-workflow modeling based on BO

A meta-model of workflow based on BO is proposed, as shown in Figure 5. The workflow model can be defined as follows.

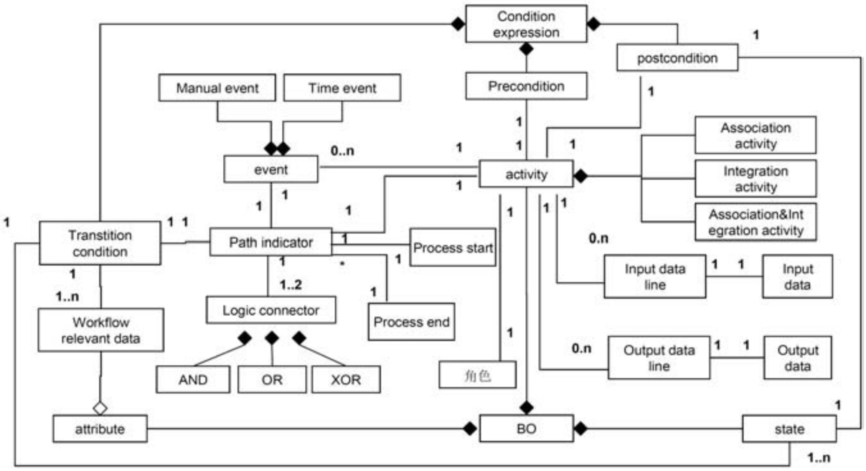
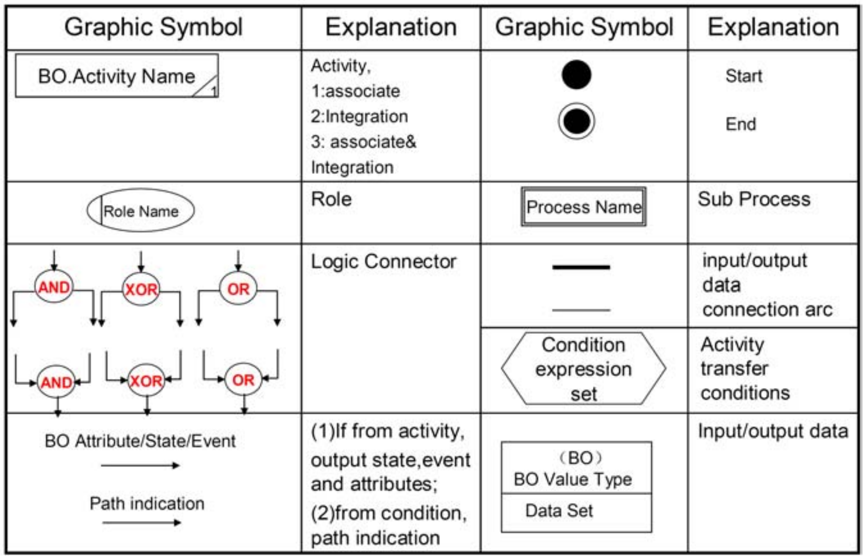


Figure 5. Meta model of workflow based on business object.

Definition 2 (Workflow model)**.** Workflow model can be noted as $WfM = (BO, Act, \prec, T, Cond, C, R, RD, S, E)$, where BO is a set of BOs in business process, $Act = \{bo_i.act_j | i = 1, \dots m, \ j = 1, \dots n\}$ is a set of activities, where act_j belongs to bo_i, $\prec$ is a set of temporal orders between any two activities, T is a set of state transition, $Cond$ is transition condition of WfM, involves precondition, postcondition and execution condition and event E condition, C is a set of control structures, $C = C_{\text{AND}} \cup C_{\text{XOR}} \cup C_{\text{OR}}$, R is a set of roles, RD is set of workflow relevant data, S is a set of states, and E is a set of events.

Based on the definitions of workflow model and meta-model, a platform-independent workflow modeling is proposed, called ICEWf (Interoperable Configurable Executable Workflow). The model elements are shown in Figure 6.

Model Element

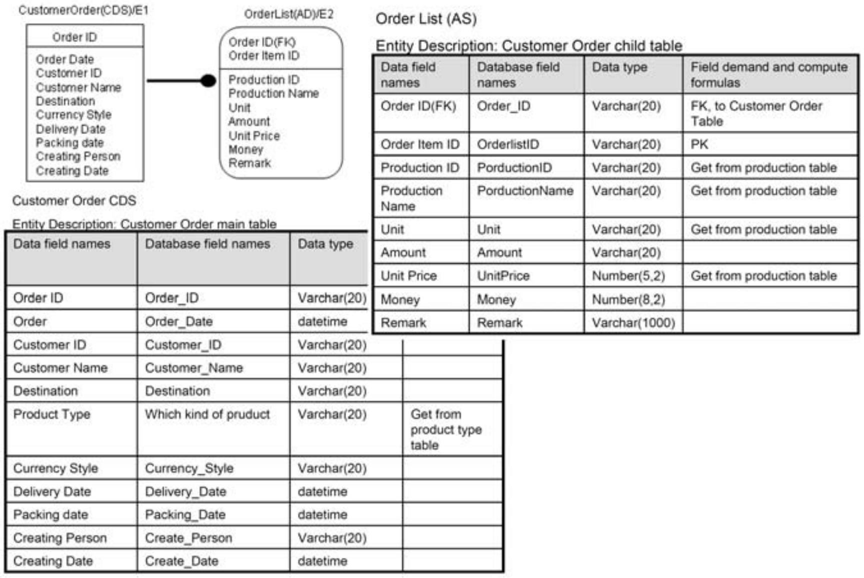


Graphic Symbol	Explanation	Graphic Symbol	Explanation
BO.Activity Name 1	Activity, 1:associate 2:Integration 3: associate& Integration	● ◉	Start End
Role Name	Role	Process Name	Sub Process
AND XOR OR AND XOR OR	Logic Connector	▬ —	input/output data connection arc
		Condition expression set	Activity transfer conditions
BO Attribute/State/Event → Path indication →	(1)If from activity, output state,event and attributes; (2)from condition, path indication	（BO） BO Value Type Data Set	Input/output data

Figure 6. Workflow modeling element.

4. Case Study

We will take a customer order as a case to show how to build the BO model and workflow model using modeling. The BO data diagram and all descriptions of each data item are shown in Figure 7.



Order List (AS)

Entity Description: Customer Order child table

Data field names	Database field names	Data type	Field demand and compute formulas
Order ID(FK)	Order_ID	Varchar(20)	FK, to Customer Order Table
Order Item ID	OrderlistID	Varchar(20)	PK
Production ID	PorductionID	Varchar(20)	Get from production table
Production Name	PorductionName	Varchar(20)	Get from production table
Unit	Unit	Varchar(20)	Get from production table
Amount	Amount	Varchar(20)	
Unit Price	UnitPrice	Number(5,2)	Get from production table
Money	Money	Number(8,2)	
Remark	Remark	Varchar(1000)	

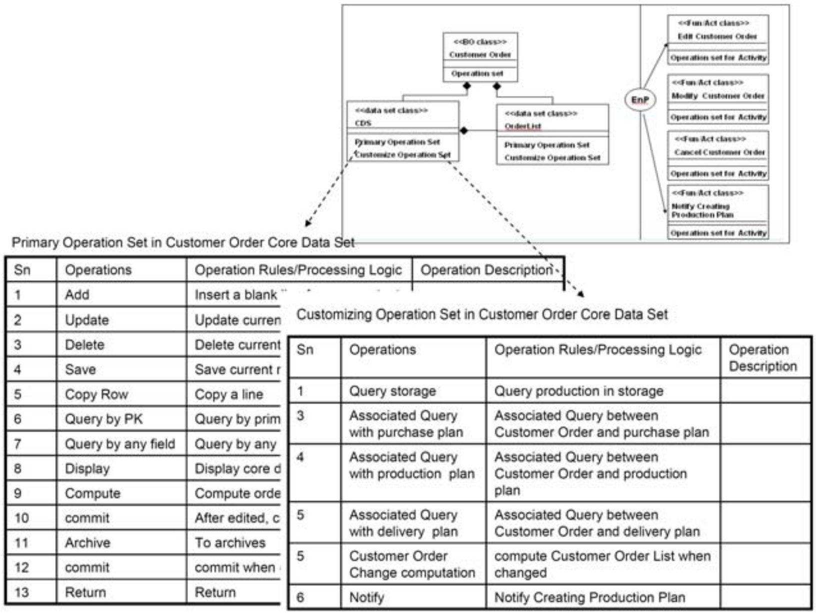
Customer Order CDS

Entity Description: Customer Order main table

Data field names	Database field names	Data type	
Order ID	Order_ID	Varchar(20)	
Order	Order_Date	datetime	
Customer ID	Customer_ID	Varchar(20)	
Customer Name	Customer_Name	Varchar(20)	
Destination	Destination	Varchar(20)	
Product Type	Which kind of pruduct	Varchar(20)	Get from product type table
Currency Style	Currency_Style	Varchar(20)	
Delivery Date	Delivery_Date	datetime	
Packing date	Packing_Date	datetime	
Creating Person	Create_Person	Varchar(20)	
Creating Date	Create_Date	datetime	

Figure 7. BO data diagram of customer order.

The BO class diagram, descriptions of primary operation set and custom operation set are shown in Figure 8. The class diagram for its order list and the activities are the same, so omitted.



Primary Operation Set in Customer Order Core Data Set

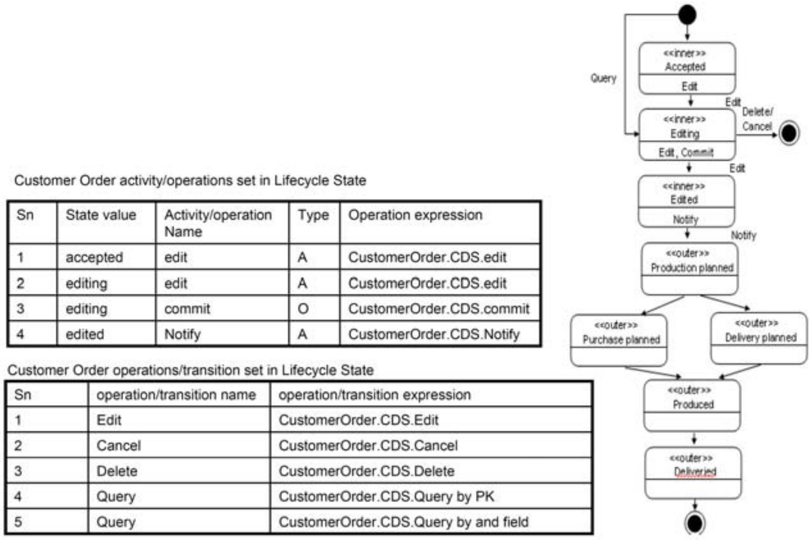
Sn	Operations	Operation Rules/Processing Logic	Operation Description
1	Add	Insert a blank	
2	Update	Update curren	
3	Delete	Delete current	
4	Save	Save current r	
5	Copy Row	Copy a line	
6	Query by PK	Query by prim	
7	Query by any field	Query by any	
8	Display	Display core d	
9	Compute	Compute orde	
10	commit	After edited, c	
11	Archive	To archives	
12	commit	commit when	
13	Return	Return	

Customizing Operation Set in Customer Order Core Data Set

Sn	Operations	Operation Rules/Processing Logic	Operation Description
1	Query storage	Query production in storage	
3	Associated Query with purchase plan	Associated Query between Customer Order and purchase plan	
4	Associated Query with production plan	Associated Query between Customer Order and production plan	
5	Associated Query with delivery plan	Associated Query between Customer Order and delivery plan	
5	Customer Order Change computation	compute Customer Order List when changed	
6	Notify	Notify Creating Production Plan	

Figure 8. BO class diagram of customer order.

The BO state diagram and the descriptions of all data items for customer order are shown in Figure 9.



Customer Order activity/operations set in Lifecycle State

Sn	State value	Activity/operation Name	Type	Operation expression
1	accepted	edit	A	CustomerOrder.CDS.edit
2	editing	edit	A	CustomerOrder.CDS.edit
3	editing	commit	O	CustomerOrder.CDS.commit
4	edited	Notify	A	CustomerOrder.CDS.Notify

Customer Order operations/transition set in Lifecycle State

Sn	operation/transition name	operation/transition expression
1	Edit	CustomerOrder.CDS.Edit
2	Cancel	CustomerOrder.CDS.Cancel
3	Delete	CustomerOrder.CDS.Delete
4	Query	CustomerOrder.CDS.Query by PK
5	Query	CustomerOrder.CDS.Query by and field

Figure 9. BO state diagram of customer order.

The BO workflow diagram is shown in Figure 10.

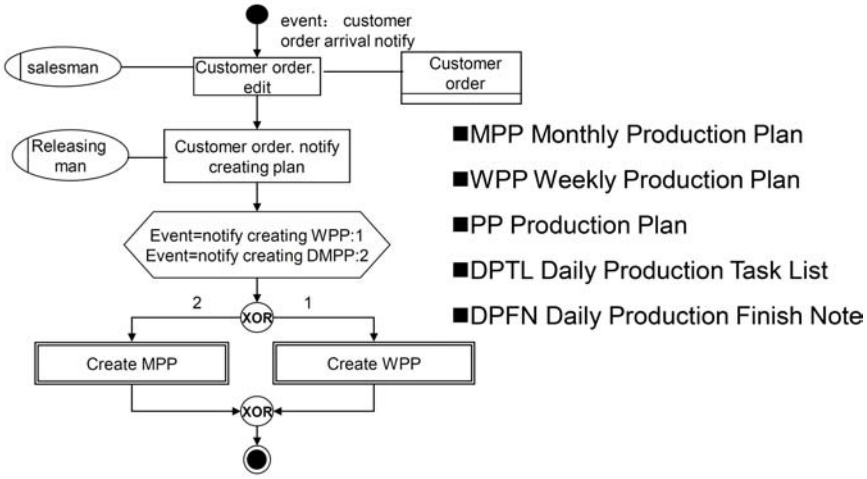

event：customer order arrival notify
salesman
Customer order. edit
Customer order
Releasing man
Customer order. notify creating plan
Event=notify creating WPP:1
Event=notify creating DMPP:2
2
XOR
1
Create MPP
Create WPP
XOR
■MPP Monthly Production Plan
■WPP Weekly Production Plan
■PP Production Plan
■DPTL Daily Production Task List
■DPFN Daily Production Finish Note

Figure 10. BO workflow model of customer order.

Two sub workflow models in customer order process are shown in Figure 11. They are the activity *monthly production plan* and the activity *weekly production plan*.

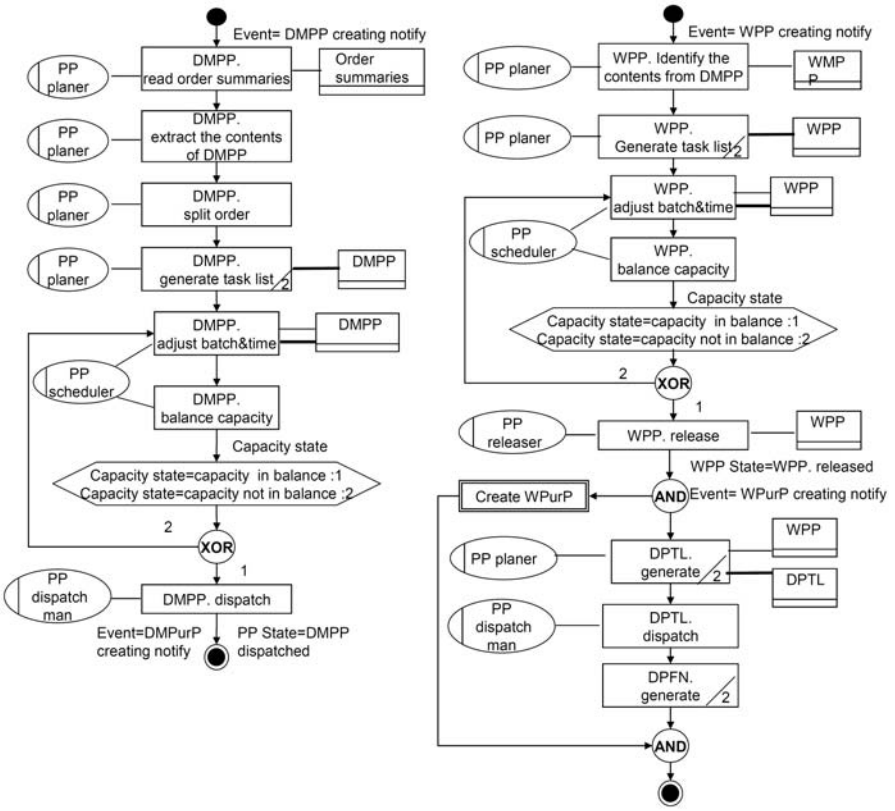


Figure 11. Two sub-worlflow models of customer order.

5. Results and Discussion

5.1. Comparisons with meta-model of WfMC

By comparing meta-model of ICEWf with that of WfMC, as shown in Table 1, we conclude that most of the elements in ICEWf have the same semantic expression with those in WfMC. Because the conception of BO is introduced, activities and states are all object-oriented. This characteristic makes a great improvement in the flexibility of ESA. Some modeling elements support changeability of business processes, e.g., classifying activities by association relation between BOs, considering states of BO as workflow relevant data. Moreover, by introducing the conception of BO, activity and other data-related elements can all be mapped to software component, which belongs to PSM level in MDA.

Table 1. Comparisons with meta-model of WfMC

	Meta-model by WfMC [8]	ICEWf
Activity condition	An independent logic step including flow condition, execution condition, and notification condition	An independent logic step, which is encapsulated in BO precondition and post-condition are the same as flow condition by WfMC.
Workflow relevant data	Provided to workflow engine for scheduling	Attributes and states of BO which are provided to workflow engine for scheduling.
Logic connector	Including sequence, fork, join, decision, merge	Same semantic expression.

5.2. Interoperability with UML

ICEWf has capability to interoperate with other models via UML. The conversation of modeling elements from ICEWf diagram to UML diagram can be seen in Figure 12.

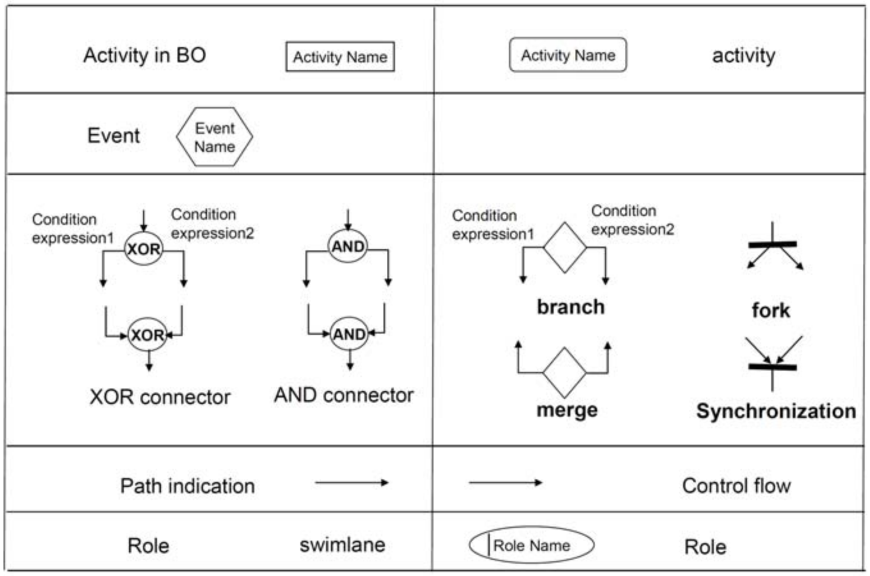


Figure 12. Elements in ICEWf and UML.

The semantics of all elements are same except the element *event*, which has no matching element in UML. Actually, it is equivalent in UML that when an activity finishes, the next activity will be started without waiting for any event.

6. Conclusion

In this paper, platform-independent modeling method, ICE-PIM is proposed to improve the agility of ESAs. The method aims to shift the focus of iterative development to a higher level of abstraction. ICE-PIM can achieve this because it is object-oriented. Moreover, BO model and workflow model represent the characteristic of ESAs, and can be transformed to executable code conveniently. ICEWf, as a workflow model whose elements are extracted from BO, follows the criterion of both WfMC and object-oriented, so that it is regarded as more flexible modeling method. The proposed approach, as well as a corresponding code generator, has been applied to modeling business in some enterprises for many years, and has a good performance.

Acknowledgements

The authors are grateful to Proc. Dechen Zhan, Research Centre of Intelligent Computing for Enterprises & Service, School of Computer Science and Technology, Harbin Institute of Technology. This work was supported by Quanzhou Science and Technology Plan Projects (Nos. 2015Z125).

References

[1] L. Zouhaier, Y. B. Hlaoui and L. Jemni Ben Ayed, Generating Accessible Multimodal User Interfaces using MDA-Based Adaptation Approach. Proc. 38th IEEE Int'l Conf. Computer Software and Applications Conference (COMPSAC), (2014), 535-540.

[2] H. I.-Ching, T. Der-Hong and H. Nien-Lin, MDA-based visual modeling approach for resources link relationships using UML profile, Computer Standards & Interfaces 36(3) (2014), 648-656.

[3] Zonghua Li, Xiaofeng Zhou, Aihua Gu and Qinfeng Li, A complete approach for CIM modelling and model formalising, Information and Software Technology 65 (2015), 39-55.

[4] Yasmina Rahmoune, Allaoua Chaoui and Elhillali Kerkouche, A framework for modeling and analysis UML activity diagram using graph transformation, Procedia Computer Science 56 (2015), 612-617.

[5] David Ameller, Xavier Burgués, Oriol Collell, Dolors Costal, Xavier Franch and Mike P. Papazoglou, Development of service-oriented architectures using model-driven development: A mapping study, Information and Software Technology 62 (2015), 42-66.

[6] Z. M. Ma, Li Yan and Fu Zhang, Modeling fuzzy information in UML class diagrams and object-oriented database models, Fuzzy Sets and Systems 186(1) (2012), 26-46.

[7] D. Maplesden, E. Tempero, J. Hosking and J. C. Grundy, Performance analysis for object-oriented software: A systematic mapping, IEEE Transactions on Software Engineering 41(7) (2015), 691-710.

[8] L. Sabari Chandramohan, G. Ravikumar, S. Doolla, and S. A. Khaparde, Business process model for deriving CIM profile: A case study for Indian utility, IEEE Transactions on Power Systems 30(1) (2015), 132- 141.

[9] P. Coad and E. Yourdon, Object-Oriented Analysis (2nd Edition), Prentice Hall, 1991, pp. 59.

[10] A. Dennis and B. H. Wixom, Systems Analysis and Design, John Wiley & Sons, Language, Addison-Wesley, 2000, pp. 45-55.

[11] Chung-Yang Chen, Gen-Yih Liao and Ku-Shen Lin, An attribute-based and object-oriented approach with system implementation for change impact analysis in variant product design, Computer-Aided Design 62 (2015), 203-217.

[12] Antoni Olivé and Ruth Raventós, Modeling events as entities in object-oriented conceptual modeling languages, Data & Knowledge Engineering 58(3) (2006), 243-262.

[13] Thomas Thüm, Christian Kästner, Fabian Benduhn, Jens Meinicke, Gunter Saake and Thomas Leich, Feature IDE: An extensible framework for feature-oriented software development, Science of Computer Programming 79 (2014), 70-85.

[14] MC. Workflow Management Coalition Specification: Terminology & Glossary, Document Number WFMC-TC-1011, Brussels, 1996.

[15] Jeong Seok Kang and Hong Seong Park, Automatic generation algorithm of expected results for testing of component-based software system, Information and Software Technology 57 (2015), 1-20.

■