An Information System Development Method Based on the Link of Business Process Modeling with Executable UML Modeling and its Evaluation by Prototyping

Tsutomu Okawa[†], Tsukasa Kaminishi[†], Syuichi Hirabayashi[‡],
Hisao Koizumi[‡], Jun Sawamoto[†††]
[†]Mitsubishi Electric Corp. Information Technology Center
[‡]Tokyo Denki University [†††]Iwate Prefectural University
Okawa.Tsutomu@ay.MitsubishiElectric.co.jp

Abstract

Currently, Business Process Modeling (BPM) is used to create current analytical model (As-Is model) of the work processes, upon which an optimized model (the To-Be model) can be based, including the optimization of system configuration. Also, there is strong interest in the eXecutableUML (xUML), which makes it possible to validate system viability before it is implemented. This paper proposes a methodology for configuring information systems that links BPM and xUML techniques, and evaluates its applicability. In the proposed method, bottlenecks are analyzed and evaluated using simulation of the work process flow in BPM and, after translating this model into UML model elements, and after the UML model has been created, expands it to xUML. Then, in xUML modeling, dynamic system evaluation is carried out with automatic generation of the code for actual implementation. The proposed method was applied to the development of an actual prototype information system and evaluated.

1. Introduction

In recent years, attention has been focused particularly on Business Process Modeling (BPM)[1] in upper processes. BPM models the contents of work as business processes, and visualizes the models to facilitate efforts to improve business processes more efficient. There have been a number of researches on BPM[2][3], but there are almost no reports of research of information processing covering from the definition of requirements to implementation.

Unified Modeling Language (UML) has also been used as a modeling technique for analyzing and designing many systems[4]. UML provides standard description methods for modeling system structure and behavior, and its ability to describe systems in multi-faced way using nine diagrams is an advantage that has secured its widespread use. However, the development method based on UML at present lacks a means of validating the correct operation of source code until the implementation is over, so that it remains unclear whether the model formed will completely satisfy the initial requirements and whether the code based on the model so formed is correct.

Currently, eXecutable UML (xUML)[5] is attracting attention as a means of solving these problems of UML by making it possible to describe behavior executable within the model. xUML was proposed by the Object Management Group (OMG)[6] and it is one of the techniques for implementing Model Driven Architecture (MDA)[7] for which specifications are currently being prepared. In MDA, a platform-independent model is first created , and then transformed into a platform-dependent model for the implementation.

If, by using these modeling techniques in development, the process model confirmed by BPM could go on to use xUML for system design and validation, then we could expect further improvements in the efficiency of information system development.

This is the background to the authors' research which, by using BPM for work process analysis, transforming the results into a UML model, and then using xUML to confirm its execution, has achieved implementation in a linked method for system development[8][9]. However, the prior research had not established the procedures for using BPM, and there were no guidelines to the transformation from BPM to UML models, and none had been taken as far as implementation after system validation using xUML.

The paper proposes a development method of information systems that links BPM with xUML. The method uses BPM to improve the work process, transforms it to UML, and expands it to xUML. Then, xUML is used to validate system operation by the simulation developed in the research, and finally code is automatically generated for implementation.

The method goes beyond prior work in detailing the modeling stages in the BPM process, and sets up modeling guidelines for the UML transformation of model elements. Also xUML was used to validate the system and to check execution.

In order to evaluate this method, it was applied to the creation of a prototype information system for ticket reservations operating over a network.

2. Issues Faced by Previous Development Methods

The conventional development method raises the following issues. Because the work and the system are so closely interconnected, the system needs to be configured closely in accordance with the work. Usually, in analyses performed as a part of the development method, it is difficult to identify the items for systemization, and to analyze work that includes these items. The work is analyzed before system design begins, and the decisions on exactly how system development should be performed become very important. Also, although the UML model created is turned into development design drawings, even if implementation is in strict accordance with these design drawings, it is not at all certain that the system implementation will, in fact, satisfy the development requirements or its specifications. So the need may arise for a redesign after implementation. This is because it is difficult to determine whether the design drawings were correct until after they were implemented.

3. A Development Method Linking BPM and xUML

3.1 The Flow of Development for the Method

In this method, BPM is used in the process of analysis for system development. The development method proceeds as a design stage using UML is followed by a verification stage using xUML. This method largely breaks down into three processes: the BPM process, the UML process and the xUML process. In the BPM process, work analysis is performed and the results are used to define the actual range of systemization. In the UML process, the results of BPM analysis are used to create the UML for system design drawings. And in the xUML process, the dynamic UML so created is used to create specially expanded xUML and the system dynamics are verified. The detailed process flow for this method is show in Fig. 1.

(1) The development requirements for improvements of current work and the addition of new operations, etc., are defined.

(2) In the BPM process, analysis of and improvements to the operating organizational configuration and the business processes to be systematized are performed. First, the organizational configuration and the business processes are modeled. The resulting model integrates the existing organizational configuration and processes, as As-Is model. At the same time, the system requirement specifications are prepared. To evaluate the viability of the As-Is model organization, and whether personnel are allocated appropriately, simulation is performed for

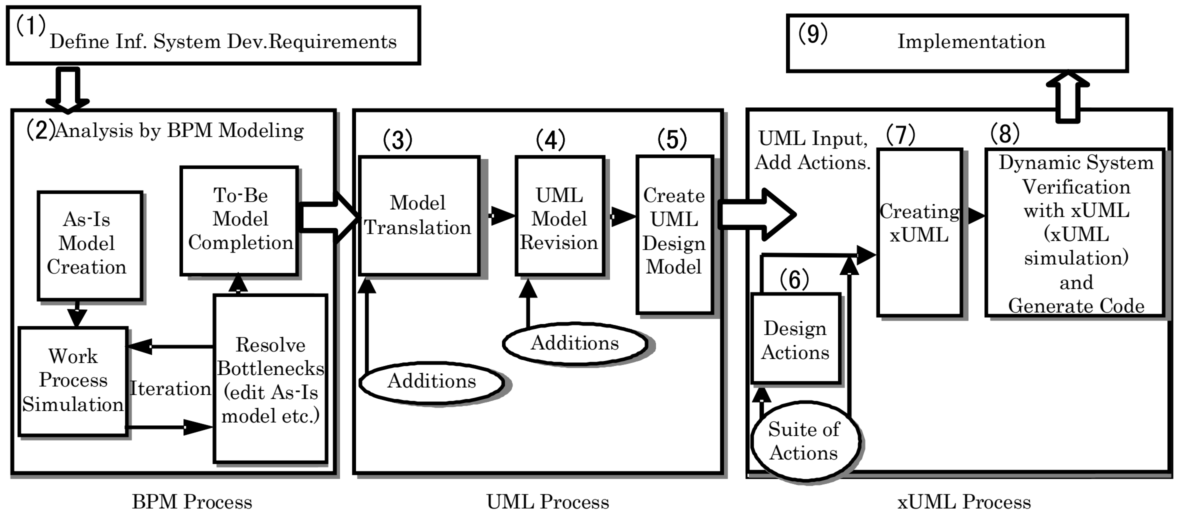


Fig. 1. The proposed development method using BPM and xUML

the As-Is model. When the As-Is model simulation reveals procedural bottlenecks, staff are re-allocated and/or the business processes are revised to improve the model in the simulation. If simulation confirms that the bottlenecks have been alleviated, the model concerned is defined as the To-Be (improved) model.

(3) Next, in the UML process, the designer adds the parameters necessary for UML to the To-Be model so created, and transforms these into UML model elements.

(4) Based on the UML model elements generated by the transformation , the UML model is modified.

(5) The UML analytical model is refined to create the UML design model.

(6) In the xUML process, the design-level UML created is evaluated for its appropriateness as a design drawing. Dynamic system simulation is performed using xUML. This is where the actions needed to expand UML to xUML are designed. Actions display the behavior of objects. In this method, it is the state chart of the UML model that is converted to xUML.

(7) The actions designed are added to the UML and xUML is created.

(8) The resulting xUML is entered in the simulator. Based on the results of the simulation, the UML design model is remade among other attempts to make improvements. The xUML simulation enables the generation of code for functions of which the design viability is confirmed.

(9) The system is implemented using the code generated.

3.2 Creating the Business Process Model

3.2.1 Modeling Guidelines.

(1) Modeling should be Performed in Successively More Detailed Stages

In modeling information systems, because it involves comparing and visualizing the relationship between the organizational structure and the system specifications, the person doing the analysis has a great deal of freedom, and modeling is strongly dependent upon the skill and experience of the modeling analyzer. This suggests that a method involving successive clarifications of the structure is required.

(2) In the BPM Process, Detailed Separation should be Made into Different Domains, and Each Domain's Sub-domain should be equivalent to a Single Business Process Model.

In the present method, the operational unit in BPM is the domain. Initially, there is a domain for the work of each department. At the stage when the business processes are being described for BPM, these domains are split into multiple sub-domains. If the domain granularity is coarse, BPM will involve many business processes straddling multiple domains and imposing a very large burden at the design stage. This is why we start by using finer domain granulation. Next, a model is created for the business process of each domain or sub-domain, and relationships established between the models such that the sub-domains form the structural hierarchy of their domain. This splitting into small units, because it increases the independence of the domain, enables us to treat a domain as a single component, and offers encouraging prospects of improving the degree of re-use and the ease of maintenance.

(3) The Business Process of Each Sub-domain should be Displayed Within a Single Screen.

In this method, a single business process view is transformed only into a single UML model. The process of transformation from business process view to UML is semi-automatic in order to reduce the burden of having to determine on each and every occasion which elements are necessary for a particular UML. The justification for this restriction is that in implementing our approach to transformation, it is good to have fewer, independent, business process functions to transform.

3.2.2 Business Process Simulation.

In the business process simulation, the descriptions of business processes that were created are used in simulations for a set virtual time period in which the process is initiated, and the results for the total processing time, the waiting time before the next process could begin, and the conditions of the execution of each process are derived. When operations for which the process is subject to delays or when there are many work procedures that cannot be completed within the period of the simulation, the designer designates such processes as bottlenecks.

3.2.3 Transformation into UML Model Elements.

UML model element transformation is used to ensure that the UML model accurately reflects the business process analysis. In our method, ARIS[10] tools are used to make the transformation. The model elements of each business process view are translated on a one-to-one basis into the different UML diagrams. For example, take the class diagram: the elements that show the "processes" of the business process view correspond with the "methods" in the class diagram.

3.3 Development of a Simulator to Implement the xUML Process

3.3.1 Characteristics. The xUML simulator we developed has the following characteristics.

- It Provides Warnings of the Appropriateness of the Validity of the Design Processes Created

Warnings are given when the simulator judges that an error may have been included in the simulation flow. It monitors flow, and when the same flow has been repeated numerous times, generates a warning. In such cases, warnings are given even when the designer has not deliberately included error conditions in the description.

- xUML simulation provides virtual operation of the actual system to be implemented on the basis of UML, thus validating the design drawings.
- Action Mapping to Generate xUML

Rather than using small-scale programming in an action language, action mapping to UML is used to define the xUML. Also, by introducing the concept of attributes for each action, similar actions can be given different behaviors by assigning them a different attribute.

3.3.2 Simulator Configuration. The simulator is configured as shown in Fig. 2. The actions created for this set of actions are added to UML, and those expanded to xUML are read into the input of the simulator. Next, in accordance with instructions input to the control section via the user interface for interpretive execution of the actions, the simulation is executed. The action information, etc., necessary for simulation is read in the extractor section, and the simulation engine that performs the simulation sends the results in real time to the output section, informing the user. After the simulation is completed, the cumulative results are returned to the user.

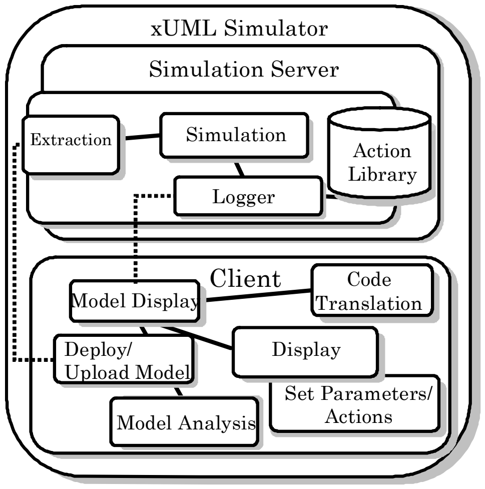


Fig. 2. The structure of xUML tools

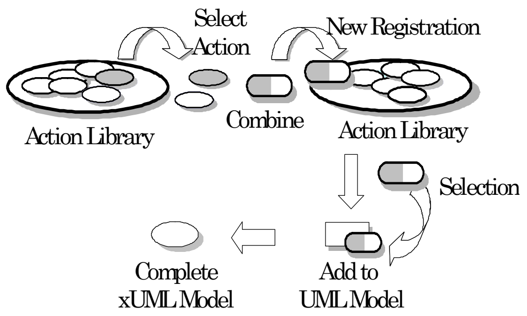


Fig. 3. Generation method using linkage of existing actions

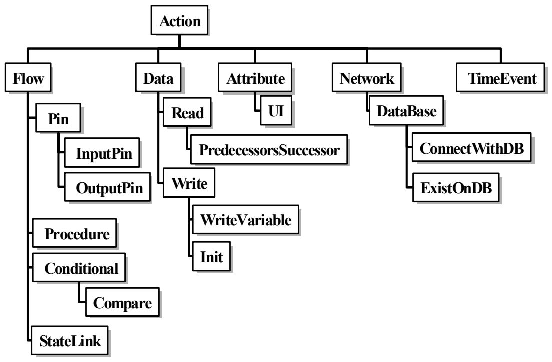


Fig. 4. The stipulated action suite

3.3.3 Action Design. The procedural flow for action design in our method is as shown in Fig. 3. In Fig.3, new actions are generated as combinations of existing actions which were developed in this research.

The actions stipulated in Fig. 4 are formed based on behaviors using the action semantics[13] produced by the Object Management Group (OMG) [6].

3.3.4 Simulator Operation. The xUML on which this xUML tool operates is a state chart. The state chart uses arrows to show the flow. By using these arrows, the simulator of our xUML tool follows and simulates the flow of the state chart that has been read into it. Fig. 5 shows the operational flow of the simulation for this system configured from the several state charts.

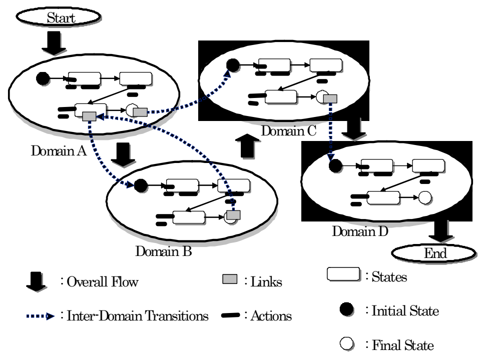


Fig. 5. The operation of xUML simulation

(1) The simulator identifies the initial state within the xUML model, and starts the simulation.

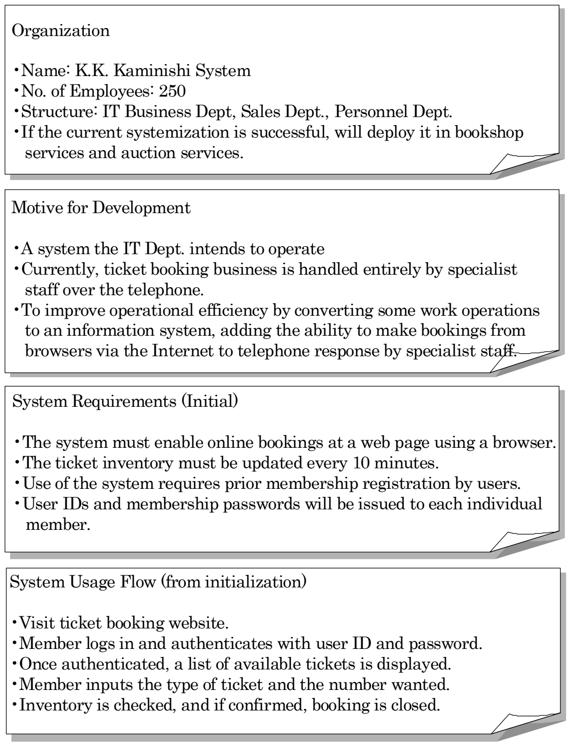
(2) The simulator follows the flow of the state chart, checking the next state, and in the process identifying the actions.

(3) The simulator conveys the actions it has identified to the simulation engine, where the actions are interpreted and executed.

(4) The results of execution are output by the output section, and the designer checks whether system operation satisfies the specifications and investigates the results for the test data that are output in the output section. Simulation, (2) and (3) above, is repeated for a single xUML model until the simulator discovers the final state.

With multiple xUML models, the "link" attribute is used to change the origin (from) and destination (to) of the link and so enable the simulation to be applied to different objects.

3.3.5 Automatic Code Generation and Implementation. The code generated is ready for execution, and there is no need for further manual refinement. This is because the execution engine uses the simulation engine itself to provide the functions necessary for implementation, and uses the model for which simulation has been completed.



Organization

- Name: K.K. Kaminishi System
- No. of Employees: 250
- Structure: IT Business Dept, Sales Dept., Personnel Dept.
- If the current systemization is successful, will deploy it in bookshop services and auction services.

Motive for Development

- A system the IT Dept. intends to operate
- Currently, ticket booking business is handled entirely by specialist staff over the telephone.
- To improve operational efficiency by converting some work operations to an information system, adding the ability to make bookings from browsers via the Internet to telephone response by specialist staff.

System Requirements (Initial)

- The system must enable online bookings at a web page using a browser.
- The ticket inventory must be updated every 10 minutes.
- Use of the system requires prior membership registration by users.
- User IDs and membership passwords will be issued to each individual member.

System Usage Flow (from initialization)

- Visit ticket booking website.
- Member logs in and authenticates with user ID and password.
- Once authenticated, a list of available tickets is displayed.
- Member inputs the type of ticket and the number wanted.
- Inventory is checked, and if confirmed, booking is closed.

Fig. 6. Application example for development

Table. 1. The Tools Forming the Proposed System and their uses

Tool	Main Functions	Dev.	Process
ARIS 6.2 Collaborative Suite	Business Process Modeling Business Process Simulation UML element translation	IDS Sheer	BPM
Enterprise Architect (EA) Corporate License	UML Modeling XMI XMI translation	Sparks Systems	UML
XUML simulator & execution environment (XDS)	xUML Action design xUML Simulation Code generation & to provide execution environment	Authors	xUML

4. Application to Prototyping and Results

4.1 The Object System of Prototyping

The proposed method was applied to the development of prototyping of a ticket reservation system for the evaluation. Fig. 6 shows an example of a company organization and development requirements.

4.2 Implementation Environment

For the BPM process of this method, in describing business processes, in the simulation of the processes created and in transforming them into UML elements, we used the ARIS[10] tools.

The applicable evironment for the prototyping is listed in Table 1, which includes the ARIS tools.

4.3 Results of the Application

4.3.1 The BPM Process Stage. The business process model created as As-Is model using BPM with the proposed method is shown in Fig. 7.

According to the modeling guidelines, the service that each department deals with was subdivided into sub-domains by the function view and a process view about each sub-domain was created. A process view was made as to fit in a screen as shown in Fig. 7. In addition, a data view that shows data treated in the process is related with the process view. In a process view in Fig. 7, user data is related as data view to the process.

Work process simulation for the resulting model of Fig. 7 used the conditions of Fig. 8. The simulation was performed for the number of processes unfinished in the period.

In the process flow of the As-Is model shown in Fig. 7, the time parameters set for processes by reception specialist staffs were modified to those of automatic telephone guidance system and the hearing process by the staffs was eliminated. Furthermore, reception

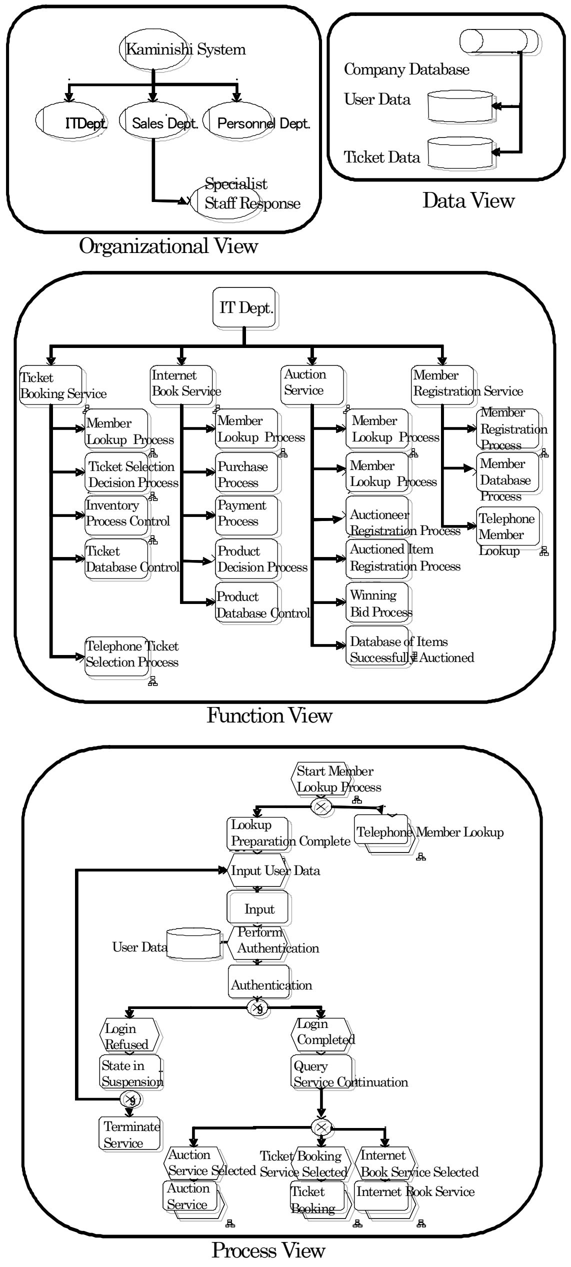


Fig. 7. Result of Modeling

Simulation Period: One week
No. of Generated Order Processes: 100,000
(Half via Internet, half by telephone)
No. of staff: 10
Processing Ability of Specialist Staff Response: 5min per process
Inventory Checks: Once for 10 minutes, 20sec per process

Fig. 8. Work Process Simulation Conditions

specialist staffs were deleted from the organization chart. The changed process flow were decided as To-Be model.

The results of a simulation after this change are shown in Fig. 9.

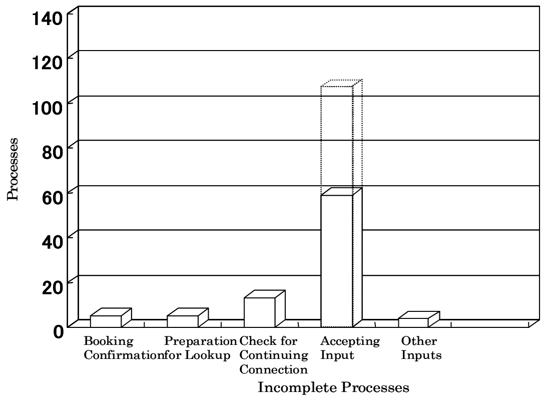


Fig. 9. Results of Simulation

4.3.2 Linkage with UML. For the business process view of Fig. 7, the transformtaion of UML elements to a state chart is shown in Fig. 10.

Fig. 10 shows the result of model conversion from the eEPC chart of Fig. 7 into a state chart. The eEPC chart of Fig. 7, created according to the modeling guidelines discussed in 3.2 and based on the domain division, fits to a screen. Owing to this model conversion, a designer was able to make the dependent UML model that was not complicated.

The generated state chart of Fig. 10 is refined at the design stage, extending it to object behavior. These behaviors are guidelines for the design of actions at the subsequent xUML stage.

4.3.3 The xUML Process Stage. Here, the actions (the actions stipulated in Fig. 4) which were developed and refined for the xUML simulator were added to the UML model, and the model was expanded into xUML. Fig. 11 shows the created xUML model. Next, xUML simulation was performed for the xUML model. The conditions for the simulation are shown in Fig. 12. The text data for the xUML simulation was taken as ticket data and user data, with flow verified from the state chart based on the scenario of Fig. 12.

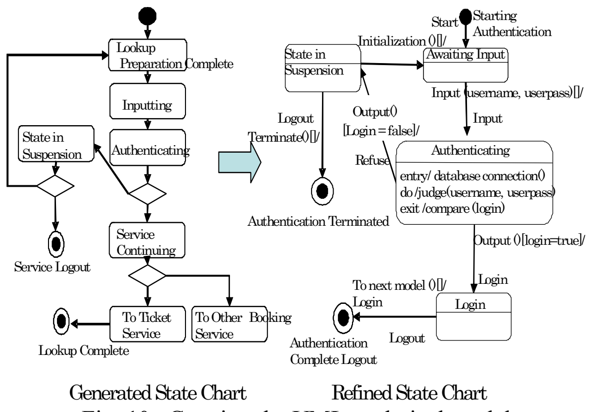


Fig. 10. Creating the UML analytical model

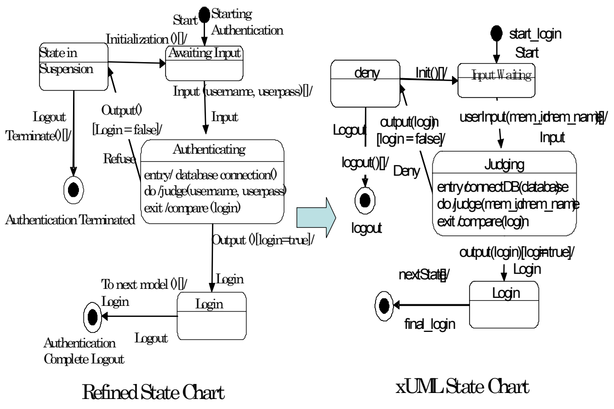


Fig. 11 Creating the xUML model from the UML design

```
Object Simulate:
  Member lookup process xUML status chart
Type:
  Scenario verification.
  Verifying judgment whether to accept or
  deny member lookup.
```

Fig. 12. xUML Process Simulation Conditions

Table. 2. Content of Log (member lookup process)

Class	Login Class
Execution state	login_state
No. of actions No. of warnings	7 1
Warning action Warning content	compare() Possible redundant action in exiting action compare()

As shown in Fig. 2, the result of xUML simulation revealed a redundant process before system implementation. The redundant process was revealed because the simulator detected a loop in the flow and gave the alarm. The redundancy in this case arose in the description given when UML was being refined. The redundant process was the "comparison" behavior for the "authenticating" state as shown in the refined state chart in Fig. 10. However, it also became clear that all processes other than this were being executed normally.

4.3.4 Code Generation and Implementation.

Next, after simulation was completed, code was generated automatically and implemented by the xUML simulator where the code for the servlets was generated to be directly deployed and executed using the MVC(Model-View-Controller) model.

5. Evaluation and Considerations

5.1 Evaluation and Considerations for the Individual Processes

(1) The BPM Process

In the BPM process, successive stages of modeling were performed, and the detailed functions in the modeling process were led to further refinements in accordance with the guidelines for the proposed method. Differing from previous design methods using modeling using UML model from the start, any UML model that satisfies Use-Case functions for the entire system was not needed. This enabled the designer to concentrate on implementing a single function at a time. The feasibility of the entire set of system functions could be verified by using business process simulation.

(2) The UML Process

The method linking the model generated by UML transformation to the UML process eliminated the occurrence of design defects in the BPM analysis of work procedures, particularly gaps arising in movements from one process to the next. However, at this stage, the UML model is completed, and it is the easiest stage at which design errors can be incorporated because it is the stage at which behaviors are created. In fact, the prototype design contained a description of a redundant process involved in member look-up in the UML state chart.

(3) The xUML Process

We confirmed the verification of system dynamic operation after reading in the state chart we had prepared and before it was implemented.

Using the action design approach of the proposed method, no additional small-scale programming was needed in action design. However, we identified an issue, that we had to repeate more than once to secure the required actions in the process of combining them because the proposed method realize the behavior by the combination of actions.

Also, in this xUML simulation a redundant process was detected in the descriptions of the UML process (5.1(2)) above. The possibility of such a redundant process in the model was clearly given in a warning by the simulator. The warning of this possibility enabled an improvement to be made before actual implementation. This is evidence for the effectiveness with which reworking after implementation can be controlled.

(4) Code Generation and Implementation

We were able to check the operation of an actual system implemented in automatically generated code. This offers support for reductions in programming costs by configuring a system without manual programmings.

5.2 Overall Considerations

The proposed method, using one simulation in the BPM process and a second in the xUML process, verified the business process model and the xUML model, and it also made possible system implementation of the code generated from the model so created. We were able to confirm the effectiveness of the method in this application as follows.

(1) It was shown to be effective in embodying improvements in system development that analysis of working operations, and reflection of the results appropriately in the development process, detected system operational bottlenecks.

(2) It was shown to be effective in detecting development errors that would otherwise have persisted, in detecting work procedural bottlenecks, and in detecting redundant processes introduced in the system design.

(3) The xUML model resulting from the completed simulations was used to generate code, and was shown to be effective by verifications of the dynamic operation of a system implemented from this code.

However, the following matters were revealed in the evaluation and remained call for further investigation.

- More number of functions of detective warning have to be added to xUML simulation. Currently, redundant processes and inappropriateness in UML descriptions are indicated as possible errors, and it has to be needed to increase the number of variations to provide support to designers.
- Currently, the xUML simulation provides few actions, so there are models that cannot be simulated and systems that cannot be configured. It has to be needed to increase the number of actions including web services to expand the coverage of xUML simulation.

6. Conclusions

In this paper, a development method of information systems that links BPM with xUML has been proposed.The method has been evaluated effectively using both the simulation in BPM and xUML simulation by the prototyping of ticket reservation system. The results of the evaluation of the method revealed the possibility of the seamless development that covers everything from defining requirements to actual implementation.

We are going to continue the following study in future.

(1) As stated in Chapter 5, we expand the coverage of xUML simulation by substantiating the function of detective warning and increasing the number of actions.

(2) We extend the variety and scale of prototype and carry out more evaluation.

(3) Based on the current study, we intend to establish a seamless development process which includes requirement definition and analysis, conversion to UML, xUML simulation, and source code generation and implementation.

References

[1]I.Junichi and T.Yasuichi :"Business Process Modeling", JUSE,(2000) (in japanese)

[2]H.Smith and P.Fingar : Business Process Management: The Third Wave,Meghan Kiffer Pr(2002)

[3]T.Kobayashi and N. Komoda : "Business Process Design Method Based on Business Event Model for Enterprise Information System Integration" ,T.IEE C,Vol.124,No.5 pp.1068-1075 (2004) (in japanese)

[4]UML,http://www.omg.org/technology/documents/formal/uml

[5]S.Mellor,M.J.Balcer : "Executable Uml: A Foundation for Model-Driven Architecture" ,Technologic Arts Inc. (2003) (in japanese)

[6]OMG : http://www.omg.org/

[7]M.Yamada : " Introduction to Model-driven Development" , IPSJ Vol.45,No1 pp3-9 (2004) (in japanese)

[8]S.Kitajima, Y.Nakamura, T.Kaminishi, H.Koizumi and K.Ban : "A Method of Information System Development based on Business Process Modeling",IPSJ DPS-119 Report,vol.2004,No.89 pp.15-20 (2004) (in japanese)

[9]Y.Kojima,S.Kitajima,T.Kaminishi,H.Koizumi and K.Ban : "An Implementation of A Parts Procurement Support System based on Business Process Modeling and Its Evaluation" ,IPSJ IS-90 Report, Vol.2004,No.116 pp.9-16 (2004) (in japanese)

[10]ARIS,http://www.ids-scheer.co.jp/,IDS Scheer

[11]EnterpriseArchitect,http://sparxsystems.jp/

[12]XMI,http://www.omg.org/technology/documents/formal/xmi.htm

[13]Alcatel, I-Logix, Kennedy-Carter, Kabira Technologies Inc., Project Technology Inc, Rational Software Corporation and Telelogic A B : ActionSemantics for the UML,OMG Report, (2001)