Workflow Process Definition and Their Applications in e-Commerce

Yen-Liang Chang[1], Sammy Chen[2], Chyun-Chyi Chen[1], Irene Chen[3]
[1]*National Central University, CSIE, Taiwan*
[2]*Software Methodology Laboratory of Academia Sinica*
[3]*Hsing Kuo University*
E-mail : chang@se01.csie.ncu.edu.tw

Abstract

Recently many enterprises face the pressures, such as market competition, reduction of cost and rapid development of new products and services. The workflow technique provides a new mechanism to help enterprises manage business processes efficiently. In the age of electronic commerce, workflow has played an important role that provides back-end services to response front-end requirements. The workflow technique reduces the time of process, allocates resource powerfully and improves the performance of enterprises. Furthermore, process definition is the core of workflow. Process definition defines all of necessary information related to business process and executed by the workflow management system. Therefore, the integrity and accuracy of process definition will affect the result of execution.

In this paper, we propose UML(Unified Modeling Language) and Petri Nets approach for workflow modeling. Our approach consists of four phases: (1) process requirement elicitation phase; (2) logical design phase; (3) properties analysis phase; (4) physical deployment phase.

1 Introduction

Workflow management system is a software system that defines, coordinates, manages, and executes complex business activities. Many enterprises are facing pressures from market competition, reduction of cost, and rapid development of new products and services, they need new techniques to reduce processing time, allocate resource efficiently, improve performance, and shorten product's time to market. Workflow management techniques give the answer. Workflow management system supports large and heterogeneous distributed execution environments where sets of interrelated tasks can be carried out in an efficient and closely supervised fashion.

With the evolution of the computer technology, workflow has experienced numbers of shifts in changes. In the early years, the process specification of workflow was hardcoded into application programs in order to satisfy certain requirements of office procedures. Nowadays, thanks to the progress of communication, information and object-oriented technologies, workflow management system has been able to support decentralized organizational units through graphical interfaces and a workflow engine to manage distributed tasks and resources on different locations [9]. Workflow has played an important role that provides back-end services to response front-end requirements in the age of electronic commence. Therefore, more and more venders have invested in the development of workflow products. These includes ActionWorkflow System of Action Technologies; IBM's Flow Mark; Visual WorkFlow of FileNet; InConcert produced by Xsoft (a division of Xerox Corp); FormFlow of Delrina; Regatta of Fujitsu (currently incorporated into ICL's TeamWARE); SAP Business Workflow by SAP; HP's WorkManager; OPEN/workflow of WANG and so on. However, many products are incompatible and no standards to enable these workflow products to work together. In 1993, the Workflow Management Coalition (WFMC) was established to encourage the development of workflow. WFMC provides a common "Reference Model" of workflow management systems to identify workflow management system's characteristics, terminology and components, and also enables individual specifications to be developed within the context of an overall model for workflow systems [10].

0-7695-0933-9/00 $10.00 © 2000 IEEE

Thus, all workflow products can achieve a level of interoperability through the use of common standard for various functions. Most related researches of workflow could be classified into process definition modeling and analysis, activity coordinating and scheduling, workflow system architecture and design, and development methodology. Of all the workflow techniques, process definition is one of the kernel parts of workflow techniques. It defines necessary information related to business process, such as the information of starting and completing conditions, constituent tasks, rules for navigating between activities, user tasks to be undertaken, applications that may be invoked and relevant data that may need to be referenced, and the resulting process definition will be executed by the workflow management system. Thus, the integrity and accuracy of process definition will affect the result of execution. We will address our UML and Petri Nets approach for workflow process definition in the following.

The rest of the paper is organized as follows. We will express what UML and business process is in Section 2. Section 3 represents how to utilize our UML approach to model business processes. In Section 4, we propose a Petri Nets model derived from activity diagram to verify properties of a process definition. We conclude this paper with our future research in Section 5.

2. UML and Business Process

In software life cycle, analysis phase is a major period to determine whether the software is corresponded to requirement of users. If the gap between domain experts or users and system developers is very narrow, the software system can be implemented to conform to the requirement of users. However, it is difficult to achieve the goal in the past. Because when the system developers receive the domain experts' description, they have trouble to catch the meaning of the terminology used by domain experts. Then, the system developers use their own system specification, such as specific specification language or unfriendly graphic representation used another terminology form a technical perspective. Further, the discrepancy from analysis to implementation results in that software system becomes difficult to use finally.

2.1 UML

In order to eliminate the difference between the business description and the software specification, unearthing common language understood by users and developers is imperative. Each symbol and semantic within the language must be defined clearly and intuitive for users. UML (Unified Modeling Language) is a well-defined and standard modeling language. UML consists of use case, sequence, collaboration, class, object, state, activity, component, and deployment diagrams [7]. A system could be modeled via these diagrams form various aspects, such as structural, behavior, implementation, and environment views. Developers can design and exchange meaningful models without losing any information, adopted by software industry. UML furnishes users with user-friendly visual notations that improve the communications between users and developers, and translate the requirements of users into software specifications more precisely. UML even provides a unified development framework from analysis phase to implementation phase with software specifications. UML exhibits rich and expressive notations and semantic for specifying, visualizing, constructing and documenting software systems, business modeling and other non-software systems. Within UML modeling elements, some extended and tailored notations are suitable to represent the process definition of workflow. In the following, we will illustrate how to make use of UML approach to specify process definition. Firstly, we adopt use case diagram to express the specification of business functionality, goals, responsibility and iterations. Secondly, we adopt class diagram to express the organization of information related to business process. Thirdly, we adopt activity diagram to model business logical steps and dynamic behavior derived from previous use case diagram. Finally we adopt deployment diagram to model the physical configuration of activities. Before starting any steps of modeling, knowing what business processes can be implemented though workflow management techniques is necessary. Next, we would like to discuss what kinds of business process are suitable for workflow.

2.2 Business Process

In workflow management systems, business process is a set of one or more procedure(s) or activity(s), which realize business objectives or policy goals such as an insurance claims process, an order process, or a loan process. Even though workflow management techniques are able to reduce manual efforts and to provide enterprises with automatic environments, but these techniques may not be suitable for all business processes. Since the concept of workflow is originally used in solving management problems of business processes, it is adapted for a business process, whose activities are allocated, scheduled, routed, managed, and executed automatically. Business processes suitable for workflow management are usually characterized with properties such as automation, monitoring, repeatability, predictability, integration, and so on. In contrast, workflow management mechanism will not be suitable for business process characterized with simple, rarely used, or needs many manual works.

Once we can identify business processes suitable for workflow management techniques, the next issue is to decide using which method and how to model these business processes. In the following, we will present our UML through an illustration of a Business-to-Customers (B2C) e-commerce order processing flow.

3. Defining Business Process Using UML

We used to transform business processes directly into logical steps, such as Petri Nets, event flow, state transition diagram, etc [1,4]. However, once processes change, we don't understand how logical steps are derived from the specification of a business process. Because such approaches only represent logical steps and lack for antecedent documentation. Therefore, in order to solve the above problems, we adopt UML approach consisting of use case, class, activity, and deployment diagrams to model diverse perspectives of business processes. We will illustrate our UML approach through an order processing example.

3.1 Use Case Diagram and Business Processes

In order to capture the context of a business process, use case diagram is useful to represent goals, responsibility, functionality, and boundary intuitively. Use case diagram also expresses static interactions between business processes and their external objects. When notations of use case diagram maps into workflow mechanism, use case notations stand for sub-processes of a business process, and actor notations stand for participants [11]. Therefore, based on the internal functions of a business process, each use case notation describes a sub-process, which composes the whole business process. Each use case also can be further detailed in another use case diagram. An actor of use case diagram may be a user, an invoked application, a database, or a legacy system. Besides, a short textual description also helps readers understand the meaning of each use case and actor. Figure 1 shows a use case diagram for an order processing. This diagram contains four use cases and five actors. In order to improve understanding, some textual descriptions for the content of each use case and actor are need. For example, the "Browse Products" use case refers to that a customer browses an on-line catalog and then completes his order. The "Place Order" use case refers to that the customer fills the detail order information such as payment information and shipping information.

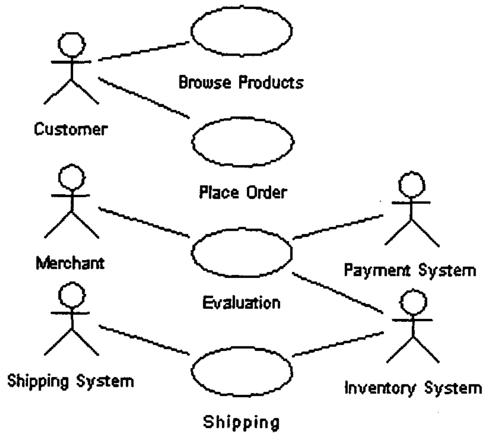

Figure 1. Use case diagram for an order processing flow

3.2 Class Diagram and Business Processes

From information aspect, class diagram of UML is useful to represent information of actors, roles, organizational units, and relevant data for business processes. These information objects can be seen as classes with relevant attributes in class diagram. In class diagram, a person may play one or more role(s) rendered by means of different classes according to his/her responsibilities. A class is an abstraction of description of a set of information objects from business processes. An attribute presents some properties within the information object and it usually displays enough information for the general readers to understand the meaning of the information object. If necessary, attributes of a class will be invoked as the process definition of the shopping is started and retrieves certain relevant data.

For example, Figure 2 shows parts of information structure of the order processing flow including six classes: Person, Merchant, Customer, Account, Order, and Product. For each class, it has its own attributes to express the intent of the class, such as the "Person" class contains id, name, and address attributes to exhibit the information of a person. No class stands alone, each works in collaboration with others to describe relevant information about business processes. Hence, Associations represent structural relationships between information objects. Association also represents both concepts of Aggregation and Generalization. These two special kinds of associations are beneficial in modeling information structure of business processes. Aggregation expresses a "whole/part" relationship, in which an information object of the whole has information objects of the part. For example, an order may have one or more product(s) shown in Figure 2. Generalization expresses an "inheritance" relationship between a general information object and a more specific information object, in which a specific information object can inherit properties of a general information object. For example, a customer or a merchant can inherit all properties defined by the "Person" class shown in Figure 2. Furthermore, association also has a multiplicity vale indicating how many instances of class may be linked to an instance of another class. For example, the "Order" class has a one-to-many association to the "Product" class referring to that at least one instance of the "Product" class is owned by one instance of the "Order" class shown in Figure 2.

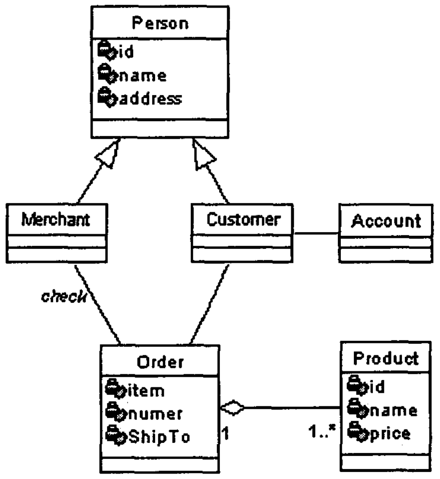

Figure 2. Class diagram for an order processing flow

3.3 Activity Diagram and Workflow Primitives

Even though use case diagram represents business processes, it cannot show the order of each use case instance and dynamic behavior. Within the UML model elements, both sequence diagram and activity diagram support to describe the dynamic behavior of use cases. Whereas sequence diagram emphasizes the flow of control from object to object, activity diagram emphasizes the flow of control form activity to activity [2]. In contrast to sequence diagram, activity diagram is very useful in modeling the process definition of the workflow and in describing the behavior that contains a lot of parallel processing. Each activity can be followed by another activity. Unlike the flow chart, the activity diagram does not only represent simply sequencing but also can direct parallel processing. This is essential for business processes. In order to improve efficiency, many tasks must be processed simultaneously within business processes.

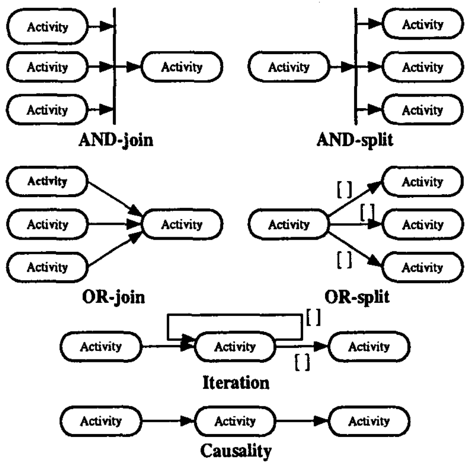

Figure 3. Workflow primitives specified by activity diagram

WFMC defined six primitives to model business logical steps [10]. In this paper, we adopt activity diagram to specify these six primitives because activity diagram supports the modeling of workflow activity, transition, condition, synchronization, parallelism, iteration, etc. We specify workflow activity by means of activity notation of activity diagram and workflow transition by means of transition notation with an arrow of activity diagram. Figure 3 shows how activity diagrams are corresponded to the six workflow primitives defined by WFMC. AND-join primitive expresses that two or more parallel threads meet into a single thread and the synchronization bar may only be crossed to next workflow activity when all input transitions on the bar have been triggered [5]. AND-split primitive expresses that a single thread split into two or more threads and the output transitions attached to the synchronization bar are triggered simultaneously. OR-join primitive expresses that when two or more alternative workflow branches re-converge into a single thread without any synchronization. OR-Split primitive expresses that when a single thread makes a decision upon which branch to take when encountered with multiple workflow branches. Branches between activities can be guarded by conditions. If guards validate, the transitions close to them are triggered to next workflow activities. Iteration primitive expresses that a workflow activity cycle involves the repetitive execution of workflow activity until a condition is met. Causality primitive expresses that two or more workflow activities are executed in a sequential form without any join or split.

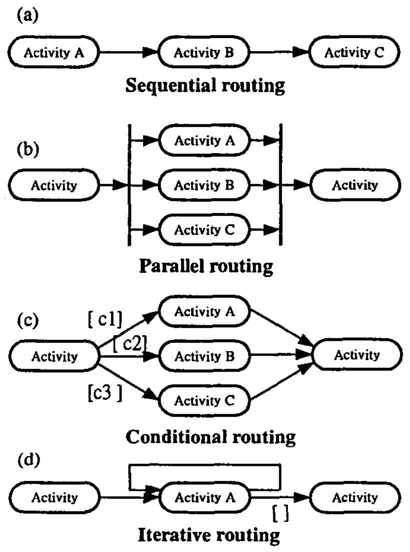

Figure 4. Process routing presented by workflow primitives

With the above six workflow primitives specified by activity diagram as shown in Figure 3, we can further to define four process routing, which are sequential, conditional, parallel, and iterative routing [1,3]. In workflow process, the four routing can be used to model any business process workflow and business process workflow can be used to model enterprise workflow. The results are show in Figure 4. Sequential routing is used to deal with causal relationships between activities. For example, three activities A, B, and C are executed sequentially. Figure 4.a shows how to use Causality workflow primitive to model sequential routing. Parallel routing is used when the ordering of activity execution is not of concern. For example, three activities A, B, and C are executed and the order of their execution is arbitrary. Figure 4.b shows how use AND-split and AND-join workflow primitives to model parallel routing. Conditional routing is used when instances need to be considered and those instances may depend on the workflow attributes. For example, in Figure 4.c one of three activities A, B, and C are executed and one of execution is depend on the workflow attributes whether satisfy condition c1, c2 and c3. Figure 4.c shows how use OR-split and OR-join workflow primitives to model conditional routing. Iterative routing is used to deal with activity which need to execute one or more than one times. Figure 4.d shows how to use iteration workflow primitive to model iterative routing.

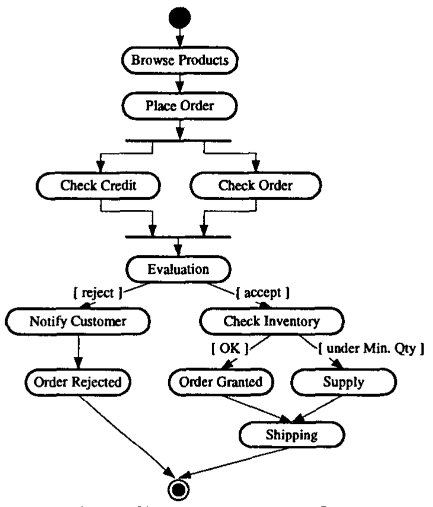

Figure 5. Activity diagram for an order processing flow

3.4 Transformation from Use Case to Activity Diagrams

Activity diagram allows the representation of logical steps of a business process for use case diagram. However, we have to know how to transform use case diagram into activity diagram. Before transformation, scenario is an advantageous mechanism to help developers understand procedures of a business process from starting to ending. Scenario is an instance of a use case that describes how use case is realized. It is a course of the flow of events for a use case, and contains preconditions, a primary scenario and one or more exceptional scenario(s). Developers can unearth objects from scenario through typical UML approach for OO modeling. In our approach, we adopt similar manners discussed previously, but we find activities applied in workflow process definition from scenarios and not objects. For example, the scenario of the "Evaluation" use case shown in Figure 1 indicates a precondition describing a placed order have been received to start with evaluation, a primary scenario describing a evaluation that the placed order and customer's credit are checked to decide whether this order is accepted or not. In the following, we will discuss the transformation of use case diagram to activity diagram.

Firstly, it is necessary to identify the preconditions of initial state and the postconditions of final state, which betters to comprehend the border of the flow of control. Then, using scenarios to work through can help identify activities of business processes and transitions from activity to activity. Before distinguishing the type of transitions, it is not hard to get sequential and branching transitions via scenarios in advance and then consider forking and joining transitions. If a transition meets a branching, guards should not overlap and must cover all possibilities. Similar to use case diagram, a complicated activity can detail further in another activity diagram. For example, Figure 5 shows the order processing flow specified by activity diagram. These activities are derived from the scenarios of use case diagram shown in Figure 1. The process starts as a customer places an order and ends as the order has been processed. It is difficult to specify forking and joining of parallel transitions at first time, such as the "Check Credit" activity and the "Check Order" activity. Hence, we can consider the flow from the "Check Credit" activity to the "Check Order" activity or from the "Check Order" activity to the "Check Credit" activity in advance and then further specify the parallel processing.

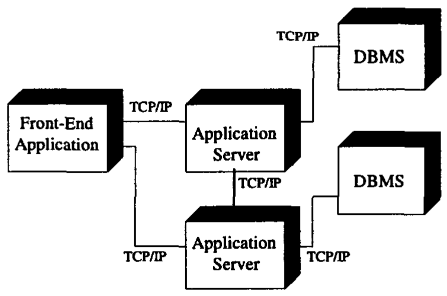

Figure 6. 3-tier deployment diagram

3.5 Deployment Diagram and Configuration

After modeling the control flow of activities, we must start to allocate these activities into physical locations. Deployment diagrams show the configuration of run-time processing elements and the software components, processes, and objects that live on then [8]. A deployment diagram is a graph of nodes connected by communication associations. Each node stands for some kind of computational unit. Nodes may contain software components that lives or runs on the node. Figure 6 shows a 3-tier deployment diagram. Thus deployment diagram is useful for modeling the physical deployment of software components in a distributed workflow management system. An activity of a business process copes with an assigned work item by executing the suitable software component on a node. Software components can be classified into a same node by component type, by functional distribution, or other partitioning mechanism.

For example, Figure 7 shows parts of the above order process in a deployment diagram. In a CORBA-based workflow management system, each node comprises an ORB, some CORBA internal services, and many software components that may be invoked applications or services. An enactment service in a single node has responsibilities for cooperating with applications to finish some work items in activities, such as "Browse Products" or "Place Order", and communicating with other enactment services in other nodes to complete remainder activities of a business process. To accomplish the target of crossing networks and platforms, CORBA-based components in distributed nodes are integrated together by using CORBA IIOP. Since components are encapsulated by IDL, application venders can use their favorite programming language to implement applications. Developers of workflow system can integrate these software components and don't have to know the implementation details.

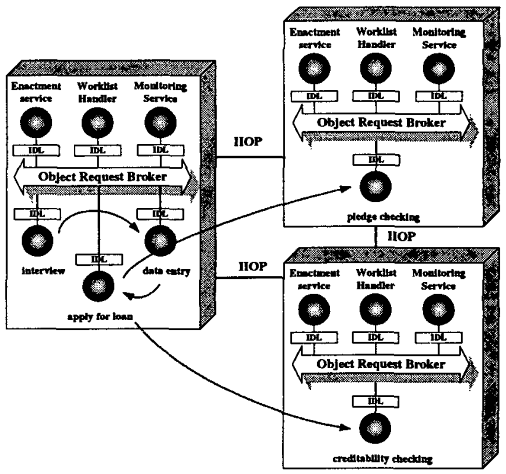

Figure 7. Deployment diagram for an order processing flow

4. Petri Nets

Petri nets (classical Petri nets) is a particular kind of directed graph [6]. A typical Petri net has transitions, places, directed arcs, and tokens in which rectangles represent transitions and circles represent places as shown in Figure 8. A transition tj is said to be enabled if each input place pi of tj has at least w(pi,tj) tokens, where w(pi,tj) is the weight of arc from pi to tj. A net is said to be an ordinary net if all of its arcs are weighted as one. In this paper, we assume all nets are ordinary. An enabled transition may fire any time when it is enabled. The firing of a transition is instantaneous and will remove one token from each of its input places and put one token into each of its output places [6]. Due to various system modeling and analysis, Petri nets have been extended to many variations, such as time Petri nets, temporal Petri nets, Coloured Petri nets (CP-nets), predicate/transition nets (PrT-Nets), hierarchical Coloured Petri nets (hierarchical CP-nets), and stochastic Petri nets.

Our approach is to transform the results of UML modeling to Petri nets, then utilize Petri nets analysis strength to complement UML's lack of analysis capability. Petri nets is a well-known modeling and analysis tool [6]. The results of Petri nets analysis are used to modify the net. The modification continues until the modified Petri nets can satisfy the specified properties. We then transform the Petri nets back to UML diagrams, and continue later phases of UML modeling.

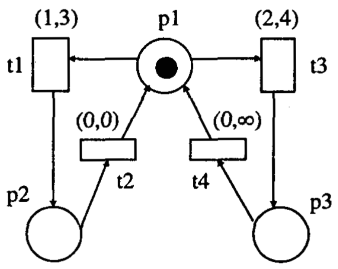

Figure 8. An example of Petri nets

4.1 Transformation of Activity Diagrams to Classical Petri Nets

In this section, we address the rules that transform activity diagrams into classical Petri nets. A state is called a source state that the state will change to other states if an event occurs. A state is called an activity state if we can further divide it into many states. A state is called an action state if we can't further divide it. Therefore, An action state is atomic and an activity state may be composed of one or more action states. In other words, an action state is a special case of an activity state. In addition, if we desire to understand the details of an activity state, we can zoom into the contents of the activity state via another activity diagram. An activity diagram would seem to be very analogous to a classical Petri net. Therefore, the suitable transformation from activity diagrams to classical Petri nets replaces the vertices of activity diagrams with transitions in Petri nets and the arcs of activity diagrams with places in classical Petri nets. The vertices of activity diagrams are represented in different ways, depending on the class of the vertices, that is, action state or branch.

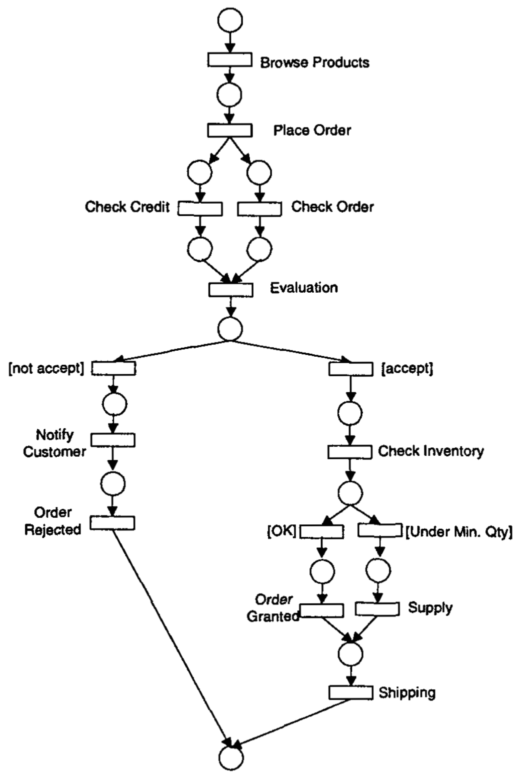

Figure 9. A Petri net model transformed from the activity diagram in Figure 5

Figure 9 shows the corresponding classical Petri net transformed from the activity diagram shown in Figure 5. Figure 10 explains how we transform activity diagrams' action state and branch vertices to Petri nets' transitions. In UML, we usually use a synchronization bar to model the forking and joining of the flows of control. In Figure 11, we explain how to transform activity diagrams' fork and join operations to classical Petri nets' transitions. Once we model business processes into a Petri nets, we can utilize many well known Petri nets analysis techniques to verify the properties of these business processes. As defined in [6], Petri nets support the specification of the system behavior and the verification of properties associated with the system behavior. We have developed a Petri nets tool for these and we use example to express it. The tool is called as NCUPN (National Central University Petri Nets toolkit). NCUPN is written in Java and available now over the Internet at http://ncupn.csie.ncu.edu.tw.

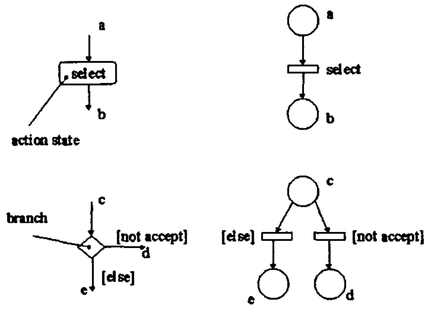

Figure 10. Transforming action state and branch vertices in activity diagrams to transitions in Peri nets

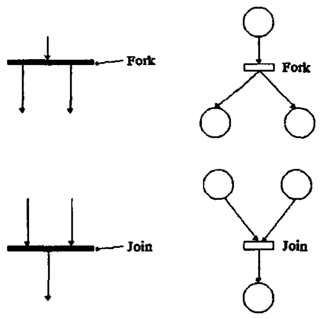

Figure 11. Transforming activity diagrams' fork and join operations to Petri nets' transitions

5. Conclusion

In this paper, we have presented an UML and Petri nets approach to model business process. We adopt use case diagram to capture the requirements of business processes, class diagram to exhibit information structure of business processes, activity diagram to express logical steps of business processes, and deployment diagram to represent physical configuration of distributed activities. All these UML modeling comply with the standard six workflow primitives defined by WFMC. Finally, we adopt Petri nets to verify the integrity and accuracy of process definition.

6. References

[1] W.M.P van der Aalst, "Three Good Reasons for Using a Petri-net-based Workflow Management System," Proc. Of the International Working Conference on Information and Process Integration in Enterprises (IPIC'96). Eds. S. Navathe and T. Wakayama, Camebridge, Massachusetts, pp. 179-201, Nov. 1996.

[2] G. Booch, J. Rumbaugh, and I. Jacobson, "The Unified Modeling Language User Guide," Addison-Wesley Longman, Inc., 1998.

[3] P. Lawrence, Workflow Management Coalition, "Workflow Handbook 1997," Wiley and Sons Ltd, New York, 1997.

[4] Y. Lei and M. P. Singh, "A Comparison of Workflow Metamodels,"http://osm7.cs.byu.edu/ER97/workshop4/ls.html, 1997.

[5] P. A. Muller, "Instant UML," Wrox Press Ltd., 1997.

[6] Murata, T., "Petri Nets: Properties, analysis and applications," in Proceeding of The IEEE, vol. 77, no. 4, pp. 541-580, 1989.

[7] Rational, and UML partners, "UML Notation Guide version 1.1," http://www.rational.com/uml/resources/documentation/notation/index.jtmp, 1997.

[8] Rational, and UML partners, "UML Summary version 1.1," http://www.rational.com/uml/resources/documentation/summary/index.jtmpl, 1997.

[9] J. Veijalanen, A. Lehtola, and O. Pihlajamaa, "Research Issues in Workflow Systems," October 2,1995.

[10] Workflow Management Coalition, "Workflow Management Coalition The Workflow Reference Model," Nov 1994.

[11] Workflow Management Coalition, "Workflow Management Coalition Terminology & Glossary," June 1996.