Chapter VII
ICN-Based Workflow Model and Its Advances

Kwanghoon Kim
Kyonggi University, South Korea

Clarence A. Ellis
University of Colorado at Boulder, USA

ABSTRACT

This chapter1 introduces the basic concepts of information control net (ICN) and its workflow models. In principle, a workflow model is the theoretical basis of a workflow modeling methodology as well as a workflow enactment architecture. Particularly, the workflow model is directly related with how its major components are embodied for implementing the underlying workflow enactment system, too. Accordingly, the authors describe the graphical and formal representations of ICN-based workflow model and its advanced models—role-based model and actor-based model—that can be automatically transformed from the ICN-based workflow model in order to improve their verifiability, maintainability and usability. Conclusively stating, we strongly believe that the ICN-based workflow model and its advanced models be very useful not only for maximizing the quality of workflows but also for strengthening theoretical backgrounds of the recent research issues, such as workflow verification/validation, workflow reengineering, workflow intelligence, workflow mining/rediscovery, and advanced workflow architectures, and so on.

INTRODUCTION

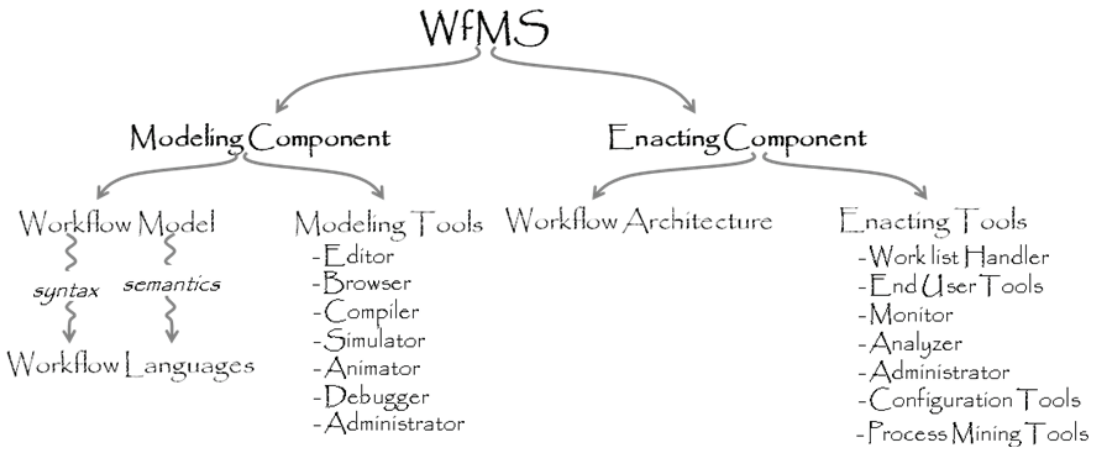
In general, a workflow management system consists of two components—modeling component and enacting component. The modeling component allows a modeler to define, analyze and maintain all of the workflow-related information which is necessary to describe a workflow procedure[2], and the enacting component supports users to play essential roles of invoking, execut-

ing and monitoring the workflow model defined by the modeling component. In other words, the logical foundation of the modeling component is the so-called workflow model that is represented by a set of entities and their relationships in order to describe a specific workflow. Therefore, the expressiveness of a workflow is decided by the underlying workflow model and its modeling system. Also, the conceptual methodology of the enacting component is called workflow architecture that is used to describing an internal structure of the workflow enacting component and characterizing its architectural style focusing on capabilities to create, manage and execute the workflow model.

In result, the workflow model and the workflow architecture become the theoretical bases on the design and implementation of a workflow management system. Of course, there may be several different types of the workflow model and different styles of the workflow architecture. According to which type of the workflow model, the underlying workflow management system may have a different way of representation and different features in modeling a workflow procedure. Likewise, different styles of the workflow architecture may show different ways of executing workflow procedures and different efficiencies as well. Therefore, the workflow model and the workflow architecture have to incorporate the advanced technological and organizational features so that the corresponding workflow management system not only displays an efficient way of modeling work and effective supports of executing performance, but also acclimates itself to a new technological and organizational environment. As the important technological trends that may affect the innovation of workflow model and architecture, we consider the powerful networked personal computing facilities like Grid/P2P computing environment, and the increasingly large and complex workflow applications; The advanced workflow models and their architectures introduced in this chapter are the outcomes of those research activities trying to improve the expressiveness of the traditional workflow model and architecture for acclimating the recent technological trends.

In this chapter, we introduce a typical workflow modeling methodology, the so-called information control net abbreviated to ICN, and describe the basic concept of ICN-based workflow model and its formalism through graphical notations and their

Figure 1. The Constituents of a Workflow Management System



formal expressions. Based upon the methodology, we also explain the detailed descriptions of the advanced workflow models and their formalisms through defining graphical and formal notations. Finally, we summarize with suggesting a list of future research and development issues related with the advanced workflow models.

BACKGROUNDS

As shown in Figure1, which is slightly modified from (Ellis & Nutt, 1993), a workflow management system is based upon two conceptual supports as well as two sets of functional supports; the former is workflow model and workflow architecture, and the latter is a set of modeling-related tools and another set of enacting-related tools. The modeling-related tools used to contain those graphical tools for a modeler to define, analyze, and maintain all the information necessary to describe a workflow procedure. In other words, in order to efficiently modeling a workflow procedure it is necessary to be supported by several tools, such as graphical editor and browser, simulator, animator, debugger and administrating tool. The workflow editor and browser take in charge of graphical editing supports to the specifications of workflow procedures, and the workflow language and its verification tools are to check the integrity of the specified workflow models. Particularly, the simulator and animator are used to checking up the pragmatical properties of the specified workflow models. Last of all, a defined and verified workflow model through those tools' supports is translated into one of the workflow languages like WPDL[3] or XPDL[4].

So far, several types of workflow models have been introduced in the literature. Almost all of the currently available workflow management systems are based upon the following types of workflow model:

- **Communication-based model (Bair, 1990).** This model stems from Winograd/ Flores' "Conversation for Action Model". It assumes that the objective of office procedure reengineering is to improve customer satisfaction. It reduces every action in a workflow for four phases based on communication between a customer and a performer: Preparation, Negotiation, Performance, and Acceptance phase. But this model is not appropriate for modeling and supporting the development of workflow implementations which have some objectives such as minimizing information system cost(not customer satisfaction), because it has some limitations in supporting the development of workflow management; For example, it is not able to indicate which activities can occur in parallel, in conditional, or in alternative.
- **Activity-based model (Ellis, 1983).** This model focuses on modeling the work instead of modeling the commitments among humans. Unlike communication-based model, activity-based model does not capture objectives such as customer satisfaction. Many commercial workflow management systems provide activity-based workflow models. The ICN-based workflow model, which is the major part of this chapter, is one of the activity-based models. Also, there are several extensions such as procedure-based model, document-based model, goal-based model, and object-oriented model. Especially, the goal-based model is a typical example that combines the communication-based model and the activity-based model.
- **Perspective-based model (Kim & Ellis, 2001).** The model supports the specification of workflows through several perspectives: the functional (the functional units of workflow processing), the behavior (the control flow of workflow), the information(the data flow of workflow), the operational(the applications deployed in workflow), and the

organizational(the organizational structure and actors who perform the workflow) perspective. This model focuses on the open aspects to support the integration of additional perspectives such as the history perspective and transactional perspective.

- **Transactional Model (Kim & Ellis, 2001).** This model involves the specification of the extended transaction management that consists of a set of constituent transactions corresponding to the workflow activities, and a set of transaction dependencies between them corresponding to the workflow structure and correctness criterion. Thus, the model focuses on the system-oriented workflows, not the human-oriented workflows. The system-oriented workflow involves computer systems that perform computation-intensive operations and specialized software tasks. That is, while the human-oriented workflow often controls an coordinates human tasks, the system-oriented workflow controls and coordinates software tasks.

The execution of a workflow model is an essential task of the enacting component. The conceptual basis of the enacting component is workflow architecture, and also its functional basis is supported by a set of enacting tools, such as worklist handler, end user (client) tools, monitoring tools, analyzer, and others. Basically, the workflow architecture is directly related with the conceptual idea in terms of how to implement an enactment controller (workflow engine) handling the execution of a workflow model, while the enacting tools have something to do with how to provide graphical and functional interfaces to the users and the invoked applications that are associated with the workflow model. The enactment controller's essential functionality is made up of information managing component and scheduling component. The information managing component's primary function is to manage the controller's database schema and also it may be strengthened with additional functions like recovering from failures without losing data or logging all execution trails and events. The scheduling component is to control the execution flow of a workflow model and to provide the communication channels among active components of the underlying workflow architecture. The control flow part is the most important factor in deciding a style of workflow architecture, and can be a decisive criterion for classifying types of workflow architecture. The followings are the important considerations characterizing the control flow part of a workflow architecture:

- What entity types of a workflow model are concretized in a workflow architecture: activity, role, actor, workcase or others
- Which of them are realized as active software modules (objects) of the workflow architecture: passive components vs. active components
- How they are communicating and interacting with each other: method invoking vs. message queueing

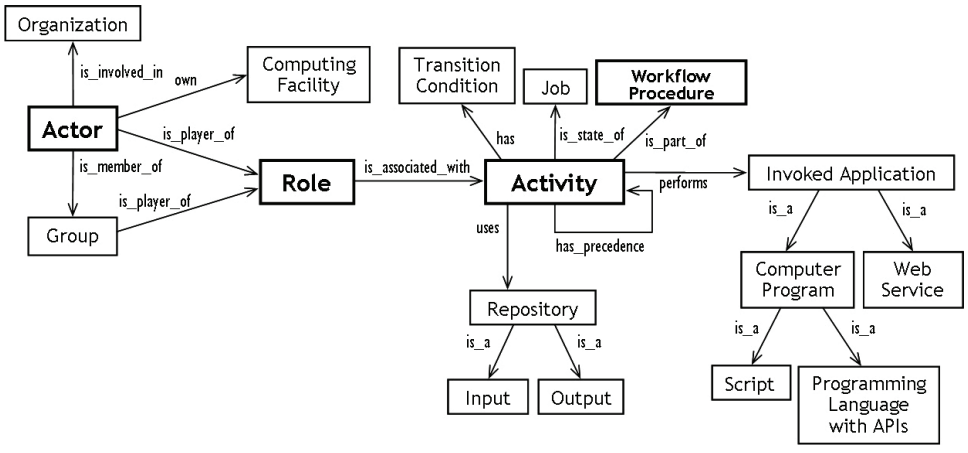
Similarly, the enacting tools primarily support two types of interactions—interactions with the end users and interactions with invoked application programs. The former is the interface with the end users through worklists assigned to each of the users, and the latter is the interface with application programs representing the specific tasks to be performed by the user. The user interface and the application program interface appear in the WfMC's reference model as worklist and invoked applications, respectively. As a result, the workflow architecture of a workflow management system is decided by the approaches that the functional supports and their interaction mechanisms are implemented. That is, the implementation architecture implies an execution infrastructure for workflow models. Also, the implementation architecture for a workflow management system can be differently configured by considering

the requirements like reliability, scalability, robustness and manageability. Therefore, it is no exaggeration to say that the efficiency of a workflow management system is determined by its underlying workflow architecture.

So far, several workflow architectures for a workflow management system have been proposed and implemented. Each of these architectures reflects a specific workflow model and contrives its own optimal implementations to satisfy various requirements from document management, imaging, application launching, and/or human coordination, collaboration, and decision making arena. However, there have been a lot of changes and evolutions in the technological computing environments and in the business circumstances as well; workflow architectures should be able to cope with not only those evolving technological trends changing from the centralized computing or the homogeneous distributed computing environments to the heterogeneous distributed computing environments without loss of performance and scalability capabilities, but also those swiftly changing business requirements from the process automation issues to the process intelligence, business activity monitoring, process agility, and the realtime enterprise architecture issues.

Conclusively, considering those advanced workflow requirements require workflow management systems to be enriched by dynamic changes, goal-based, and heterogeneous distributed and large scaled, which mean that a workflow model should provide a way to effectively represent flexibility, distribution and scalability concerns of workflow procedures; according to what workflow model is used, its workflow architecture should reflect dynamically evolving workflow procedures, distributed enactment controllers and an increasing number of actors and roles, too. However, the previous workflow models and architectures lack for supports for these advanced requirements. Therefore, in this chapter we introduce several advanced workflow models that can be feasible resolutions of those advanced requirements. We strongly believe that the advanced workflow models can be more effective and efficient if the concepts of actor, role and workcase orientations rather than activity orientation are embedded into workflow models and their workflow architectures at the same time. This chapter peculiarly addresses the Information Control Net (ICN) that is a formal methodology to systematically and formally describe the advanced workflow models, and describes their

Figure 2. The Workflow Meta-Model



implications to be expected from incorporating the advanced concepts into the advanced workflow architectures. In the following sections, we precisely state the ICN-based workflow model through an example, its meta-model and advanced workflow models. At the end, we summarize the chapter with stating the architectural implications of the ICN-based workflow model and its advanced workflow models.

ICN-Based Workflow Model

This section describes the basic concept of ICN-based workflow model (Ellis, 1983), and its advanced workflow models role-based model, and actor-based model—to be the theoretical bases for their advanced workflow architectures. In describing the ICN-based workflow model, we define its graphical notations and their formal representations. Additionally, we try to define the advanced workflow models through graphical notations and their formal representations too, and finally we specify the algorithms that are able to automatically generate the advanced workflow models from the ICN-based workflow model.

Workflow Meta-Model

In describing a ICN-based workflow model, we would use the basic workflow terminology—workflow procedure, activity, job, workcase, role, actor/group, and invoked application including web services. These terms become the primitive entity types to be composed into ICN-based workflow models, and also they have appropriate relationships with each other as shown in Figure 2. The followings are the basic definitions of the primitive entity types:

- A **workflow procedure** is defined by a predefined or intended set of tasks or steps, called activities, and their temporal ordering of executions. A workflow management system helps to organize, control, and execute such defined workflow procedures. Conclusively, a workflow procedure can be described by a temporal order of the associated activities through the combinations of sequential logics, conjunctive logics (after activity A, do activities B and C), disjunctive logics (after activity A, do activity B or C), and loop logics.
- An **activity** is a conceptual entity of the basic unit of work (task or step), and the activities in a workflow procedure have precedence relationships, each other, in terms of their execution sequences. Also, the activity can be precisely specified by one of the three entity types—compound activity, elementary activity and gateway activity. The compound activity represents an activity containing another workflow procedure, which is called subworkflow. The elementary activity is an activity that can be realized by a computer program, such as application program, transaction, script, or web service. And the gateway activity implies an activity that is used to controlling execution sequences of elementary/compound activities. The types of gateway activities consist of conjunctive gateway (after activity A, do activities B and C), disjunctive gateway (after activity A, do activity B or C), and loop gateway. Particularly, both the disjunctive gateway and the loop gateway need to be set some specific **transition conditions** in order to select one of the possible transition pathes during the execution time. The transition condition itself can be defined by using the input/output relevant data on the **repository**. Additionally, each activity has to be associated with a real performer, such as organizational staff (role, participant) and system, who possesses all ownerships over that activity.
- A **role**, as an logical unit of the organizational structure, is a named designator for one or more participants, which conveniently acts

as the basis for participating works, skills, access controls, execution controls, authority, and responsibility over the associated activity.

- An **actor** is a person, program, or entity that can fulfill roles to execute, to be responsible for, or to be associated in some way with activities and workflow procedures. Multiple instances of a workflow procedure may be in various stages of execution. Thus, the workflow procedure can be considered as a class (in object oriented terminology), and each execution, called a **workcase**, can be considered an instance. A workcase is thus defined as the locus of control for a particular execution of a workflow procedure.
- An **invoked application program** that automatically performs the associated activity, or provides automated assistance within hybrid activities are called **scripts**. If an activity is executed in automatic or hybrid mode, this means that whole/part of the invoked application program associated with the activity is automatically launched by an workflow enactment service.
- Finally, a **repository** is a set of input and output relevant data of an activity. Eventually, the repository provides a communication channel between the workflow enactment domain and the invoked application programs domain. That is, the input and the output repositories are used to realizing the input parameters and the output parameters of the associated invoked application program, respectively.

Information Control Net

An ICN-based workflow model can be defined by capturing the notations of workflow procedures, activities and their control precedence, invoked applications, roles, actors, and input/output repositories, as explained in the previous section of the workflow meta-model. In this section, we define the basic concept of workflow model with respect to the formal and graphical descriptions of ICN-based workflow model. The following [**Definition 1**] is a formal definition of ICN-based workflow model, and its functional components to be used for retrieving workflow-related information, such as activity precedence(control flow), activity-role association, activity-relevant data association(data flow), activity-invoked application association, activity-transition condition association, and role-actor association information. Based upon these types of information, it is possible to retrieve several types of derived workflow-related information like activity-actor association, relevant data-invoked application association, role complexity, actor complexity information, and so forth.

Definition 1. Information Control Net (ICN) *for formally defining workflow model. A basic **ICN** is 8-tuple $\Gamma = (\delta, \rho, \lambda, \varepsilon, \pi, \kappa, \boldsymbol{I}, \boldsymbol{O})$ over a set **A** of activities (including a set of group activities), a set **T** of transition conditions, a set **R** of repositories, a set **G** of invoked application programs, a set **P** of roles, and a set **C** of actors (including a set of actor groups), where*

- ***I** is a finite set of initial input repositories, assumed to be loaded with information by some external process before execution of the ICN;*
- ***O** is a finite set of final output repositories, perhaps containing information used by some external process after execution of the ICN;* $\delta = \delta_i \cup \delta_o$ *where,* $\delta_o : \boldsymbol{A} \rightarrow \wp\left(\wp(\boldsymbol{A})\right)$ *is a multi-valued function mapping an activity to its sets of (immediate) successors (subset of the powerset of **A**) and* $\delta_i : \boldsymbol{A} \rightarrow \wp\left(\wp(\boldsymbol{A})\right)$ *is a multi-valued function mapping an activity to its sets of (immediate) predecessors (subset of the powerset of **A**);*

$\rho = \rho_i \cup \rho_o$

where $\rho_o : A \rightarrow \wp(R)$ *is a multi-valued function mapping an activity to its set of output repositories, and* $\rho_i : A \rightarrow \wp(R)$ *is a multi-valued function mapping an activity to its set of input repositories;*

$\lambda = \lambda_\alpha \cup \lambda_g$

where $\lambda_g : A \rightarrow \wp(G)$ *is a single-valued function mapping an activity to its invoked application program, and* $\lambda_\alpha : G \rightarrow \wp(A)$ *is a multi-valued function mapping an invoked application program to its set of associated activities;*

$\varepsilon = \varepsilon_\alpha \cup \varepsilon_p$

where $\varepsilon_p : A \rightarrow \wp(P)$ *is a single-valued function mapping an activity to a role, and* $\varepsilon_\alpha : P \rightarrow \wp(A)$ *is a multi-valued function mapping a role to its sets of associated activities;*

$\pi = \pi_p \cup \pi_c$

where, $\pi_c : P \rightarrow \wp(C)$ *is a multi-valued function mapping a role to its sets of associated actors, and* $\pi_p : C \rightarrow \wp(P)$ *is a multi-valued function mapping an actor to its sets of associated roles;*

$\kappa = \kappa_i \cup \kappa_o$

where $\kappa_i : A \rightarrow \wp(T)$ *is a multi-valued function mapping an activity to a set of control-transition conditions,* ***T****, on directed arcs,* $(\delta_i(\alpha), \quad \alpha \in A)$ *between* $\delta_i(\alpha)$ *and* α*; and* $\kappa_o : A \rightarrow \wp(T)$ *is a multi-valued function mapping an activity to a set of control-transition conditions,* ***T****, on directed arcs,* $(\alpha \in A, \delta_o(\alpha))$ *between* α *and* $\delta_o(\alpha)$

Starting and Terminating Nodes. Additionally, the execution of a workflow model commences by a single χ transition-condition. So, we always assume without loss of generality that there is a single starting node ($\nabla\nabla\nabla\nabla\nabla\nabla : \alpha_1 I$) . At the commencement, it is assumed that all input repositories in the set **I** have been initialized with data by the external system:

$\alpha_1 I \in \mathbf{A} | \delta_1 i\ (\alpha_1 I\ ) = \{\emptyset\} \wedge \kappa_1 o\ (\alpha_1 I\ ) = \{\{\chi\}\}$.

The execution is terminated with any one λ output transition-condition. Also we assume without loss of generality that there is a single terminating node ($\nabla\nabla\nabla\nabla\nabla\nabla : \alpha_1 F$). The set of output repositories **O** is data holders that may be used after termination by the external system:

$\alpha_1 F \in \mathbf{A} | \delta_1 o\ (\alpha_1 F\ ) = \{\emptyset\} \wedge \kappa_1 i\ (\alpha_1 F\ ) = \{\{\chi\}\}$.

Control Flow: Temporal Ordering of Activities. Given a formal definition, the temporal ordering of activities in a workflow model can be interpreted as follows: For any activity α, in general,

$$\delta(\alpha) = \{ \{\beta_{11}, \beta_{12}, \dots, \beta_{1m(1)}\}, \{\beta_{21}, \beta_{22}, \dots, \beta_{2m(2)}\}, \dots, \{\beta_{11}, \beta_{12}, \dots, \beta_{1m(1)}\}, \}$$

means that upon the completion of activity α, a transition that simultaneously initiates all of the activities β_{i1} through $\beta_{im(i)}$ occurs, which is called a parallel transition; otherwise only one value out of $i (1 \leq i \leq n)$ is selected as the result of a decision made within activity α, which is called a decision transition. Note that if $n = 1 \wedge m = 1$, then neither decision nor parallel is needed after completion of activity α, which means that the transition is a sequential transition. Additionally stating to make sure, if $m(i) = 1$ for all *i*, then no parallel processing is initiated by completion of α.

Based on the interpretation, we graphically define these primitive transition types as shown in Figure3. The former, that an activity has a conjunctive (or parallel) transition, is represented by a solid dot(●), and the latter, that an activity has a disjunctive (or decision) transition, is represented by hollow dot(○). Besides, as defined in the previous section, these special types of

activities are called gateway activities, and in order to be syntactically safe, it is very important for these gateway activities to keep the structured properties—proper nesting and matched pair properties. Therefore, not only each of the gateway activities always keeps matched pair with split and join types of gateway activity in a workflow procedure, but also multiple sets of the gateway activities keep in a properly nested pattern. Summarily, the following is to formally describe for the basic transition types modeled by the exclusive-OR and AND gateway activities depicted in Figure 3.

Sequential Transition between activities $incoming \rightarrow \delta_i(\alpha_B) = \{\{\alpha_A\}\}$; $outgoing \rightarrow \delta_O(\alpha_B) = \{\{\alpha_C\}\}$;

Exclusive OR Transition through *xor-gateway* $xor\text{-}split \rightarrow \delta_O(\alpha_A) = \{\{\alpha_B\}, \{\alpha_C\}\}$; $xor\text{-}join \rightarrow \delta_i(\alpha_D) = \{\{\alpha_B\}, \{\alpha_C\}\}$;

AND Transition through *and-gateway* $and\text{-}split \rightarrow \delta_O(\alpha_A) = \{\{\alpha_B, \alpha_C\}\}$; $and\text{-}join \rightarrow \delta_i(\alpha_D) = \{\{\alpha_B, \alpha_C\}\}$;

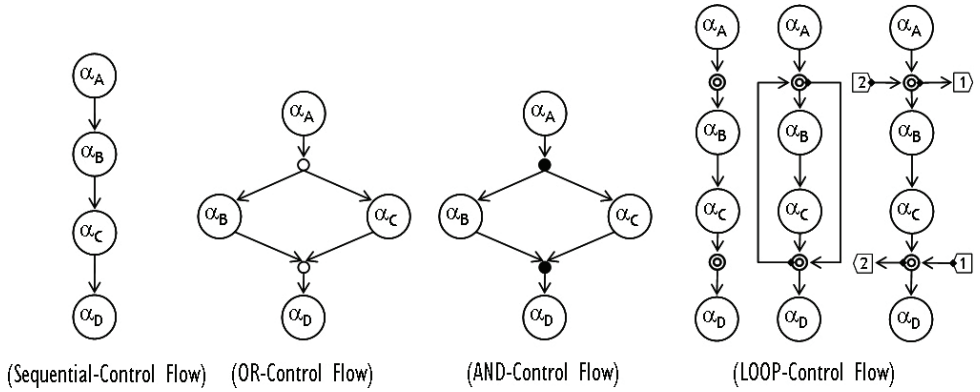
Loop Transition: Block Activity. Especially, we have to take care of an iterative (loop) transition that is the most essential as well as common construct in modeling the temporal ordering of activities. We do need to graphically define the iterative (loop) transition type as a pair of double-hollow dots of gateway activities shown in Figure 3. At a glance, it can be interpreted as a special type of disjunctive transition type; however, if we replace this transition type with a disjunctive transition type, it is very hard to maintain the structured properties—matched pair and proper nesting—of workflow model. Therefore, we introduce a concept of block[5] activity in order to keep the structured properties in modeling a workflow procedure and for the sake of the simplification of modeling work, as well. The block activity contains two gateway—loop-begin and loop-end—activities, and modeling the temporal ordering of the activities inside of the two gateway activities is done by an exactly same way of the ICN-based workflow modeling approach; justly on the gateway activities we have to specify the loop's exit conditions in the modeling time. Accordingly, the formal definition of a block activity's gateway activities shown in Figure3 is as the following:

loop-begin Gateway
$\rightarrow \delta_i(\alpha_{loop-begin}) = \{\{\alpha_A\}\}$; $\delta_O(\alpha_{loop-begin}) = \{\{\alpha_B\}\}$;

loop-end Gateway
$\rightarrow \delta_i(\alpha_{loop-end}) = \{\{\alpha_C\}\}$; $\delta_O(\alpha_{loop-end}) = \{\{\alpha_D\}\}$;

Assigning Roles and Actors. For any activity α, $\varepsilon_p(\alpha) = \{\eta_1, \eta_2, \dots, \eta_n\}$, where n is the number of roles, $\forall \eta \in P$, involved in the activity, means

Figure 3. The Information Control Net Primitives



that an activity α is performed by one of the roles; Also, $\varepsilon_\alpha(\eta) = \{\alpha_1, \alpha_2, \ldots, \alpha_m\}$, where m is the number of activities performed by the role, means that a role η is associated with several activities in a workflow procedure. Typically one or more participants are associated with each activity via roles. A role is a named designator for one or more participants which conveniently acts as the basis for partitioning of work skills, access controls, execution controls, and authority / responsibility. An actor is a person, program, or entity that can fulfill roles to execute, to be responsible for, or to be associated in some way with activities and procedures.

An Example : The Hiring Workflow Model

In order to clarify the ICN-based workflow model formally defined in the previous section, we show how a workflow procedure is transformed into the ICN-based workflow model through a hiring workflow procedure as an example. The hiring workflow procedure consists of 18 elementary activities having precedence with each other and 12 gateway activities, 10 types of roles—applicant(η_1), hiring clerk(η_2), hiring manager(η_3), personnel clerk(η_4), employment clerk(η_5), medical clerk(η_6), personnel manager(η_7), medical manager(η_8), employment manager(η_9), and computer(η_{10}), and 5 relevant data—applicant information(γ_1), decision result(γ_2), checkup done(γ_13), checkup resultγ_4 s(), and review results(γ_5)—as depicted in Figure 4. The detailed description of the elementary activities are the followings:

Elementary Activities

- The APPLY activity(α_1) is accessed by an applicant. The applicant fills out an application form through the employment page on the World Wide Web or the employment interfaces. This entails creating a workcase of the hiring procedure and starting the workcase's execution. Applicants should give the following information: personnel data, security data, affirmative action data including working preference, education, employment experience, etc.
- The NEW APPLICANT INFO activity(α_2) validates the application information written by an applicant, stores it in the database, and prepares and distributes the information for the medical screening, the security checking, and the background checking activities
- The DECISION activity(α_3) reviews and evaluates the applicant's information and decides whether the applicant is eligible and appropriate for the requirements of an open position.
- The REJECTING activity(α_4) receives the applicants who failed in the employment procedure, composes a rejection letter, and sends it to them.
- The DATABASE UPDATE activity-Rejecting(α_5) updates the employment database automatically.
- The HIRING activities$[(\alpha]_6, \alpha_7, \alpha_8)$ physically consist of three activities: request compensation(α_6), offer letter(α_7), and hiring activity(α_8). It receives the applicants who passed in the employment procedure, composes a job offer letter, and sends it to them after deciding the applicant's salary.
- The SECURITY SCREENING LOOP activities$[(\alpha]_9, \alpha_{12})$consist of checking activity(α_{10}) and reviewing activity(α_{10}), which validates the security information written by the applicants through iterations of checking and reviewing activities. After checking the information, the actor writes the checking results with comments. Then after reviewing the results, the security manager continues the security testing loop activities until the results satisfy the organization's rules and regulations.

Figure 4. The Hiring Workflow Model: ICN's Control and Data Flow

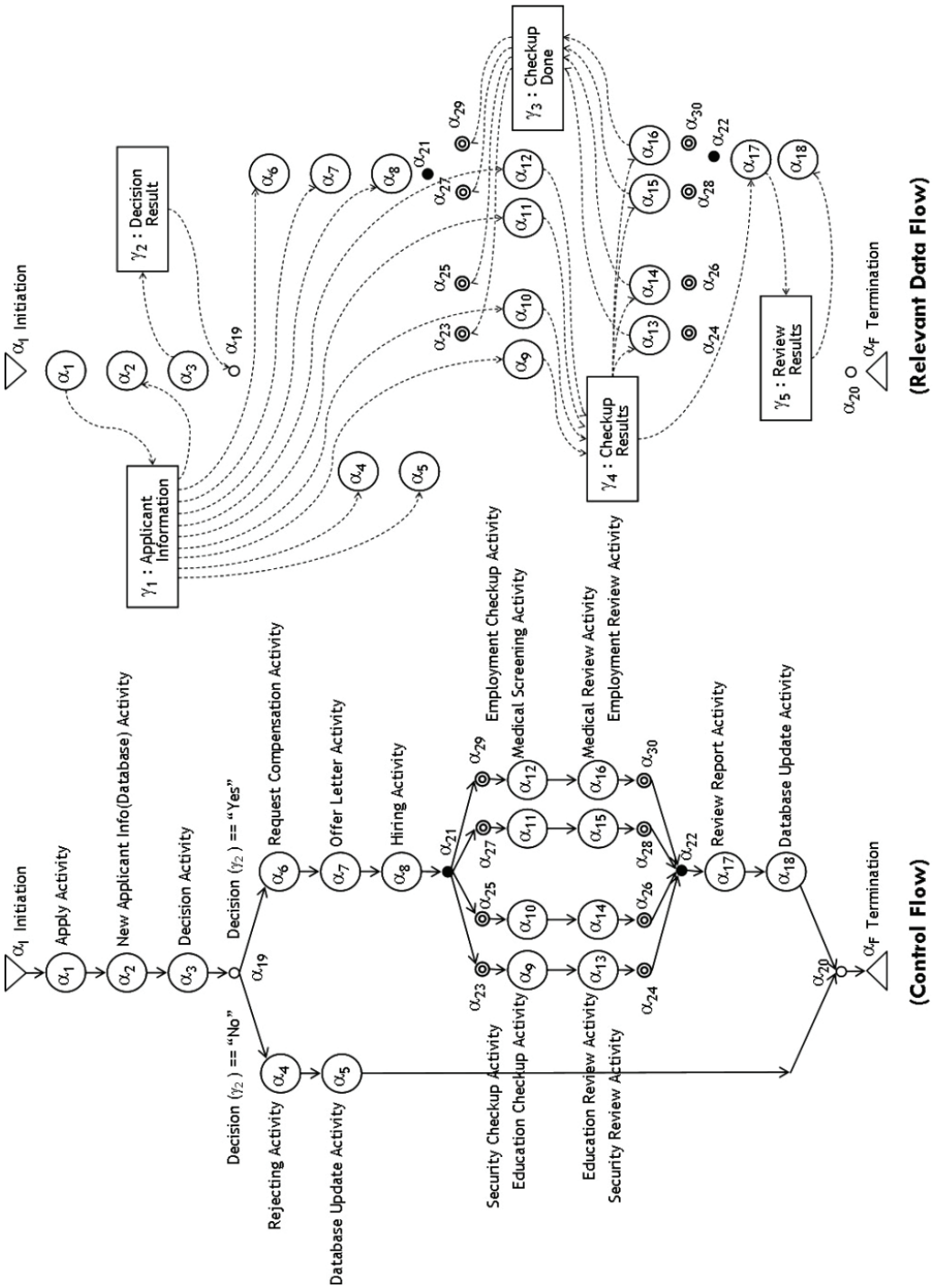


Figure 5. The Hiring Workflow Model: ICN's Role and Actor Assignments

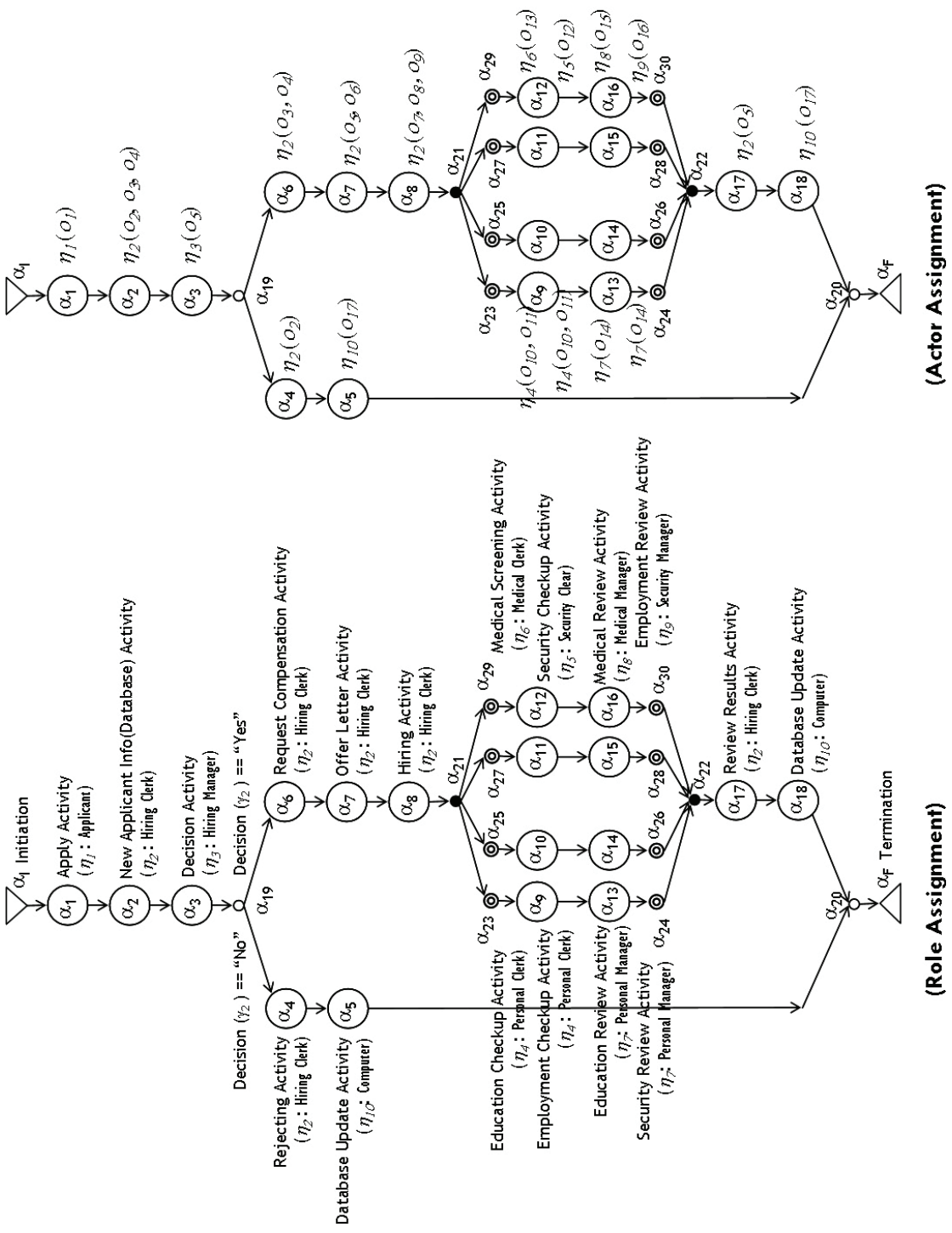


Table 1. Formal Representation of the ICN-based Hiring Workflow Model

$\Gamma = (\delta, \rho, \lambda, \varepsilon, \pi, \kappa, \mathbf{I}, \mathbf{O})$ over $\mathbf{A}, \mathbf{R}, \mathbf{P}, \mathbf{C}, \mathbf{T}$	The ICN-based Hiring Workflow Model
$\mathbf{A} = A_{elementary} \cup A_{gateway} \cup A_{block}$	Activities
$A_{elementary} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8, \alpha_9, \alpha_{10}, \alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_{14}, \alpha_{15}, \alpha_{16}, \alpha_{17}, \alpha_{18}, \alpha_I, \alpha_F\}$	Elementary Activities
$A_{gateway} = \{\alpha_{19}, \alpha_{20}, \alpha_{21}, \alpha_{22}, \alpha_{23}, \alpha_{24}, \alpha_{25}, \alpha_{26}, \alpha_{27}, \alpha_{28}, \alpha_{29}, \alpha_{30}\}$	Gateway Activities
$A_{block} = \{block_1(\alpha_{23} \triangleright \alpha_{24}), block_2(\alpha_{25} \triangleright \alpha_{26}), block_3(\alpha_{27} \triangleright \alpha_{28}), block_4(\alpha_{29} \triangleright \alpha_{30})\}$	Block Activities
$\mathbf{R} = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5\}$	Repositories
$\mathbf{P} = \{\eta_1, \eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_7, \eta_8, \eta_9, \eta_{10}\}, \eta_2 = \{\eta_{2.1} \cup \eta_{2.2} \cup \eta_{2.3} \cup \eta_{2.4} \cup \eta_{2.5} \cup \eta_{2.6}\}$	Roles
$\mathbf{C} = \{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_8, o_9, o_{10}, o_{11}, o_{12}, o_{13}, o_{14}, o_{15}, o_{16}, o_{17}\}$	Actors
$\mathbf{G} = \{\tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6, \tau_7, \tau_8, \tau_9, \tau_{10}, \tau_{11}, \tau_{12}, \tau_{13}, \tau_{14}, \tau_{15}, \tau_{16}, \tau_{17}\}$	Invoked Applications
$\mathbf{T} = \{d(default), tc_1(\gamma_2 =' No'), tc_2(\gamma_2 =' Yes'), tc_3(\gamma_3 =' No'), tc_4(\gamma_3 =' Yes')\}$	Transition Conditions
$\mathbf{I} = \{\emptyset\}$	Initial Input Repositories
$\mathbf{O} = \{\emptyset\}$	Final Output Repositories

$\delta = \delta_i \cup \delta_o$ /* Control Flow */

$\delta_i(\alpha_I) = \{\emptyset\}$; $\delta_i(\alpha_1) = \{\alpha_I\}$;
$\delta_i(\alpha_2) = \{\alpha_1\}$; $\delta_i(\alpha_3) = \{\alpha_2\}$;
$\delta_i(\alpha_4) = \{\alpha_3\}$; $\delta_i(\alpha_5) = \{\alpha_4\}$;
$\delta_i(\alpha_6) = \{\alpha_3\}$; $\delta_i(\alpha_7) = \{\alpha_6\}$;
$\delta_i(\alpha_8) = \{\alpha_7\}$; $\delta_i(block_1) = \{\alpha_8\}$;
$\delta_i(block_2) = \{\alpha_8\}$;
$\delta_i(block_3) = \{\alpha_8\}$;
$\delta_i(block_4) = \{\alpha_8\}$;
$\delta_i(\alpha_{17}) = \{block_1, block_2, block_3, block_4\}$;
$\delta_i(\alpha_{18}) = \{\alpha_{17}\}$;
$\delta_i(\alpha_F) = \{\{\alpha_5\}, \{\alpha_{18}\}\}$;
$\mathbf{A}_{\text{block1}}$:
$\delta_i(\alpha_{23}) = \{\alpha_8\}$; $\delta_i(\alpha_9) = \{\alpha_{23}\}$;
$\delta_i(\alpha_{13}) = \{\alpha_9\}$; $\delta_i(\alpha_{24}) = \{\alpha_{13}\}$;
$\mathbf{A}_{\text{block2}}$:
$\delta_i(\alpha_{25}) = \{\alpha_8\}$; $\delta_i(\alpha_{10}) = \{\alpha_{25}\}$;
$\delta_i(\alpha_{14}) = \{\alpha_{10}\}$; $\delta_i(\alpha_{26}) = \{\alpha_{14}\}$;
$\mathbf{A}_{\text{block3}}$:
$\delta_i(\alpha_{27}) = \{\alpha_8\}$; $\delta_i(\alpha_{11}) = \{\alpha_{27}\}$;
$\delta_i(\alpha_{15}) = \{\alpha_{11}\}$; $\delta_i(\alpha_{28}) = \{\alpha_{15}\}$;
$\mathbf{A}_{\text{block4}}$:
$\delta_i(\alpha_{29}) = \{\alpha_8\}$; $\delta_i(\alpha_{12}) = \{\alpha_{29}\}$;
$\delta_i(\alpha_{16}) = \{\alpha_{12}\}$; $\delta_i(\alpha_{30}) = \{\alpha_{16}\}$;

$\delta_o(\alpha_I) = \{\alpha_1\}$; $\delta_o(\alpha_1) = \{\alpha_2\}$;
$\delta_o(\alpha_2) = \{\alpha_3\}$; $\delta_o(\alpha_3) = \{\alpha_4, \alpha_6\}$;
$\delta_o(\alpha_4) = \{\alpha_5\}$; $\delta_o(\alpha_5) = \{\alpha_F\}$;
$\delta_o(\alpha_6) = \{\alpha_7\}$; $\delta_o(\alpha_7) = \{\alpha_8\}$;
$\delta_o(\alpha_8) = \{block_1, block_2, block_3, block_4\}$;
$\delta_o(block_1) = \{\alpha_{17}\}$;
$\delta_o(block_2) = \{\alpha_{17}\}$;
$\delta_o(block_3) = \{\alpha_{17}\}$;
$\delta_o(block_4) = \{\alpha_{17}\}$;
$\delta_o(\alpha_{17}) = \{\alpha_{18}\}$;
$\delta_o(\alpha_{18}) = \{\alpha_F\}$; $\delta_o(\alpha_F) = \{\emptyset\}$;
$\mathbf{A}_{\text{block1}}$:
$\delta_o(\alpha_{23}) = \{\alpha_9\}$; $\delta_o(\alpha_9) = \{\alpha_{13}\}$;
$\delta_o(\alpha_{13}) = \{\alpha_{24}\}$; $\delta_o(\alpha_{24}) = \{\alpha_{17}\}$;
$\mathbf{A}_{\text{block2}}$:
$\delta_o(\alpha_{25}) = \{\alpha_{10}\}$; $\delta_o(\alpha_{10}) = \{\alpha_{14}\}$;
$\delta_o(\alpha_{14}) = \{\alpha_{26}\}$; $\delta_o(\alpha_{26}) = \{\alpha_{17}\}$;
$\mathbf{A}_{\text{block3}}$:
$\delta_o(\alpha_{27}) = \{\alpha_{11}\}$; $\delta_o(\alpha_{11}) = \{\alpha_{15}\}$;
$\delta_o(\alpha_{15}) = \{\alpha_{28}\}$; $\delta_o(\alpha_{28}) = \{\alpha_{17}\}$;
$\mathbf{A}_{\text{block4}}$:
$\delta_o(\alpha_{29}) = \{\alpha_{12}\}$; $\delta_o(\alpha_{12}) = \{\alpha_{16}\}$;
$\delta_o(\alpha_{16}) = \{\alpha_{30}\}$; $\delta_o(\alpha_{30}) = \{\alpha_{17}\}$;

- The EDUCATIONAL BACKGROUND SCREENING LOOP activities$(\alpha_{10}, \alpha_{14})$, validates the educational background information submitted by the applicant. After checking the information, the actor prepares the checking results with some comments. Then after reviewing the results, the manager continues the educational background testing loop activities until the results satisfy the organization's rules and regulations.
- The MEDICAL SCREENING LOOP activities$(\alpha_{11}, \alpha_{15})$ do a series of medical tests, such as drugs, venereal diseases, and geriatric diseases. After testing, the actor prepares the test results with some comments, and send them to the personal department. Then after reviewing the results, the manager continues the medical screening loop activities until the results satisfy the organization's rules and regulations.
- The EMPLOYMENT EXPERIENCE SCREENING LOOP activities$(\alpha_{12}, \alpha_{16})$ validates the employment experience information submitted by the applicant. After checking the information, the actor prepares the checking results with some comments. Then after reviewing the results, the manager continues the employment experience testing loop activities until the results satisfy the organization's rules and regulations.
- The REVIEW APPLICANT INFO activity(α_{17}) reviews the results sent by the previous activities, and decides whether the applicant should be failed or passed, based

Table 2. Continuing: Formal Representation of the ICN-based Hiring Workflow Model

$\rho = \rho_i \cup \rho_o$ /* Data Flow */	$\rho_i(\alpha_1) = \{\emptyset\}$; $\rho_i(\alpha_2) = \{\gamma_1\}$; $\rho_i(\alpha_3) = \{\gamma_1\}$; $\rho_i(\alpha_4) = \{\gamma_1\}$; $\rho_i(\alpha_5) = \{\gamma_1\}$; $\rho_i(\alpha_6) = \{\gamma_1\}$; $\rho_i(\alpha_7) = \{\gamma_1\}$; $\rho_i(\alpha_8) = \{\gamma_1\}$; $\rho_i(\alpha_9) = \{\gamma_1\}$; $\rho_i(\alpha_{10}) = \{\gamma_1\}$; $\rho_i(\alpha_{11}) = \{\gamma_1\}$; $\rho_i(\alpha_{12}) = \{\gamma_1\}$; $\rho_i(\alpha_{13}) = \{\gamma_4\}$; $\rho_i(\alpha_{14}) = \{\gamma_4\}$; $\rho_i(\alpha_{15}) = \{\gamma_4\}$; $\rho_i(\alpha_{16}) = \{\gamma_4\}$; $\rho_i(\alpha_{17}) = \{\gamma_4\}$; $\rho_i(\alpha_{18}) = \{\gamma_5\}$; $\rho_i(\alpha_{19}) = \{\gamma_2\}$; $\rho_i(\alpha_{20}) = \{\emptyset\}$; $\rho_i(\alpha_{21}) = \{\emptyset\}$; $\rho_i(\alpha_{22}) = \{\emptyset\}$; $\rho_i(\alpha_{23}) = \{\gamma_3\}$; $\rho_i(\alpha_{24}) = \{\emptyset\}$; $\rho_i(\alpha_{25}) = \{\gamma_3\}$; $\rho_i(\alpha_{26}) = \{\emptyset\}$; $\rho_i(\alpha_{27}) = \{\gamma_3\}$; $\rho_i(\alpha_{28}) = \{\emptyset\}$; $\rho_i(\alpha_{29}) = \{\gamma_3\}$; $\rho_i(\alpha_{30}) = \{\emptyset\}$; $\rho_i(\alpha_F) = \{\emptyset\}$;	$\rho_o(\alpha_1) = \{\gamma_1\}$; $\rho_o(\alpha_2) = \{\emptyset\}$; $\rho_o(\alpha_3) = \{\gamma_2\}$; $\rho_o(\alpha_4) = \{\emptyset\}$; $\rho_o(\alpha_5) = \{\emptyset\}$; $\rho_o(\alpha_6) = \{\emptyset\}$; $\rho_o(\alpha_7) = \{\emptyset\}$; $\rho_o(\alpha_8) = \{\emptyset\}$; $\rho_o(\alpha_9) = \{\gamma_4\}$; $\rho_o(\alpha_{10}) = \{\gamma_4\}$; $\rho_o(\alpha_{11}) = \{\gamma_4\}$; $\rho_o(\alpha_{12}) = \{\gamma_4\}$; $\rho_o(\alpha_{13}) = \{\gamma_3\}$; $\rho_o(\alpha_{14}) = \{\gamma_3\}$; $\rho_o(\alpha_{15}) = \{\gamma_3\}$; $\rho_o(\alpha_{16}) = \{\gamma_3\}$; $\rho_o(\alpha_{17}) = \{\gamma_5\}$; $\rho_o(\alpha_{18}) = \{\emptyset\}$; $\rho_o(\alpha_{19}) = \{\emptyset\}$; $\rho_o(\alpha_{20}) = \{\emptyset\}$; $\rho_o(\alpha_{21}) = \{\emptyset\}$; $\rho_o(\alpha_{22}) = \{\emptyset\}$; $\rho_o(\alpha_{23}) = \{\emptyset\}$; $\rho_o(\alpha_{24}) = \{\emptyset\}$; $\rho_o(\alpha_{25}) = \{\emptyset\}$; $\rho_o(\alpha_{26}) = \{\emptyset\}$; $\rho_o(\alpha_{27}) = \{\emptyset\}$; $\rho_o(\alpha_{28}) = \{\emptyset\}$; $\rho_o(\alpha_{29}) = \{\emptyset\}$; $\rho_o(\alpha_{30}) = \{\emptyset\}$; $\rho_o(\alpha_F) = \{\emptyset\}$;
$\lambda = \lambda_g \cup \lambda_a$ /* Invoked Applications Assignments */	$\lambda_g(\alpha_1) = \{\emptyset\}$; $\lambda_g(\alpha_1) = \{\tau_1\}$; $\lambda_g(\alpha_2) = \{\tau_2\}$; $\lambda_g(\alpha_3) = \{\tau_3\}$; $\lambda_g(\alpha_4) = \{\tau_4\}$; $\lambda_g(\alpha_5) = \{\tau_5\}$; $\lambda_g(\alpha_6) = \{\tau_6\}$; $\lambda_g(\alpha_7) = \{\tau_7\}$; $\lambda_g(\alpha_8) = \{\tau_8\}$; $\lambda_g(\alpha_9) = \{\tau_9\}$; $\lambda_g(\alpha_{10}) = \{\tau_{10}\}$; $\lambda_g(\alpha_{11}) = \{\tau_{11}\}$; $\lambda_g(\alpha_{12}) = \{\tau_{12}\}$; $\lambda_g(\alpha_{13}) = \{\tau_{13}\}$; $\lambda_g(\alpha_{14}) = \{\tau_{14}\}$; $\lambda_g(\alpha_{15}) = \{\tau_{15}\}$; $\lambda_g(\alpha_{16}) = \{\tau_{16}\}$; $\lambda_g(\alpha_{17}) = \{\tau_{17}\}$; $\lambda_g(\alpha_{18}) = \{\tau_5\}$; $\lambda_g(\alpha_F) = \{\emptyset\}$;	$\lambda_a(\tau_1) = \{\alpha_1\}$; $\lambda_a(\tau_2) = \{\alpha_2\}$; $\lambda_a(\tau_3) = \{\alpha_3\}$; $\lambda_a(\tau_4) = \{\alpha_4\}$; $\lambda_a(\tau_5) = \{\alpha_5, \alpha_{18}\}$; $\lambda_a(\tau_6) = \{\alpha_6\}$; $\lambda_a(\tau_7) = \{\alpha_7\}$; $\lambda_a(\tau_8) = \{\alpha_8\}$; $\lambda_a(\tau_9) = \{\alpha_9\}$; $\lambda_a(\tau_{10}) = \{\alpha_{10}\}$; $\lambda_a(\tau_{11}) = \{\alpha_{11}\}$; $\lambda_a(\tau_{12}) = \{\alpha_{12}\}$; $\lambda_a(\tau_{13}) = \{\alpha_{13}\}$; $\lambda_a(\tau_{14}) = \{\alpha_{14}\}$; $\lambda_a(\tau_{15}) = \{\alpha_{15}\}$; $\lambda_a(\tau_{16}) = \{\alpha_{16}\}$; $\lambda_a(\tau_{17}) = \{\alpha_{17}\}$;
$\varepsilon = \varepsilon_p \cup \varepsilon_a$ /* Role Assignments */	$\varepsilon_p(\alpha_1) = \{\emptyset\}$; $\varepsilon_p(\alpha_1) = \{\eta_1\}$; $\varepsilon_p(\alpha_2) = \{\eta_{2.1}\}$; $\varepsilon_p(\alpha_3) = \{\eta_3\}$; $\varepsilon_p(\alpha_4) = \{\eta_{2.2}\}$; $\varepsilon_p(\alpha_5) = \{\eta_{10}\}$; $\varepsilon_p(\alpha_6) = \{\eta_{2.3}\}$; $\varepsilon_p(\alpha_7) = \{\eta_{2.4}\}$; $\varepsilon_p(\alpha_8) = \{\eta_{2.5}\}$; $\varepsilon_p(\alpha_9) = \{\eta_4\}$; $\varepsilon_p(\alpha_{10}) = \{\eta_4\}$; $\varepsilon_p(\alpha_{11}) = \{\eta_5\}$; $\varepsilon_p(\alpha_{12}) = \{\eta_6\}$; $\varepsilon_p(\alpha_{13}) = \{\eta_7\}$; $\varepsilon_p(\alpha_{14}) = \{\eta_7\}$; $\varepsilon_p(\alpha_{15}) = \{\eta_8\}$; $\varepsilon_p(\alpha_{16}) = \{\eta_9\}$; $\varepsilon_p(\alpha_{17}) = \{\eta_{2.6}\}$; $\varepsilon_p(\alpha_{18}) = \{\eta_{10}\}$; $\varepsilon_p(\alpha_F) = \{\emptyset\}$;	$\varepsilon_a(\eta_1) = \{\alpha_1\}$; $\varepsilon_a(\eta_2) = \{\alpha_2, \alpha_4, \alpha_6, \alpha_7, \alpha_8, \alpha_{17}\}$; $\varepsilon_a(\eta_{2.1}) = \{\alpha_2\}$; $\varepsilon_a(\eta_{2.2}) = \{\alpha_4\}$; $\varepsilon_a(\eta_{2.3}) = \{\alpha_6\}$; $\varepsilon_a(\eta_{2.4}) = \{\alpha_7\}$; $\varepsilon_a(\eta_{2.5}) = \{\alpha_8\}$; $\varepsilon_a(\eta_{2.6}) = \{\alpha_{17}\}$; $\varepsilon_a(\eta_3) = \{\alpha_3\}$; $\varepsilon_a(\eta_4) = \{\alpha_9, \alpha_{10}\}$; $\varepsilon_a(\eta_5) = \{\alpha_{11}\}$; $\varepsilon_a(\eta_6) = \{\alpha_{12}\}$; $\varepsilon_a(\eta_7) = \{\alpha_{13}, \alpha_{14}\}$; $\varepsilon_a(\eta_8) = \{\alpha_{15}\}$; $\varepsilon_a(\eta_9) = \{\alpha_{16}\}$; $\varepsilon_a(\eta_{10}) = \{\alpha_5, \alpha_{18}\}$;
$\pi = \pi_p \cup \pi_c$ /* Actor Assignments */	$\pi_p(o_1) = \{\eta_1\}$; $\pi_p(o_2) = \{\{\eta_2, \eta_{2.1}, \eta_{2.2}\}\}$; $\pi_p(o_3) = \{\{\eta_2, \eta_{2.1}, \eta_{2.3}\}\}$; $\pi_p(o_4) = \{\{\eta_2, \eta_{2.1}, \eta_{2.3}\}\}$; $\pi_p(o_5) = \{\{\eta_2, \eta_{2.4}, \eta_{2.6}\}, \{\eta_3\}\}$; $\pi_p(o_6) = \{\{\eta_2, \eta_{2.4}\}\}$; $\pi_p(o_7) = \{\{\eta_2, \eta_{2.5}\}\}$; $\pi_p(o_8) = \{\{\eta_2, \eta_{2.5}\}\}$; $\pi_p(o_9) = \{\{\eta_2, \eta_{2.5}\}\}$; $\pi_p(o_{10}) = \{\eta_4\}$; $\pi_p(o_{11}) = \{\eta_4\}$; $\pi_p(o_{12}) = \{\eta_5\}$; $\pi_p(o_{13}) = \{\eta_6\}$; $\pi_p(o_{14}) = \{\eta_7\}$; $\pi_p(o_{15}) = \{\eta_8\}$; $\pi_p(o_{16}) = \{\eta_9\}$; $\pi_p(o_{17}) = \{\eta_{10}\}$;	$\pi_c(\eta_1) = \{o_1\}$; $\pi_c(\eta_2) = \{o_2, o_3, o_4, o_5, o_6, o_7, o_8, o_9\}$; $\pi_c(\eta_{2.1}) = \{o_2, o_3, o_4\}$; $\pi_c(\eta_{2.2}) = \{o_2\}$; $\pi_c(\eta_{2.3}) = \{o_3, o_4\}$; $\pi_c(\eta_{2.4}) = \{o_5, o_6\}$; $\pi_c(\eta_{2.5}) = \{o_7, o_8, o_9\}$; $\pi_c(\eta_{2.6}) = \{o_5\}$; $\pi_c(\eta_3) = \{o_5\}$; $\pi_c(\eta_4) = \{o_{10}, o_{11}\}$; $\pi_c(\eta_5) = \{o_{12}\}$; $\pi_c(\eta_6) = \{o_{13}\}$; $\pi_c(\eta_7) = \{o_{14}\}$; $\pi_c(\eta_8) = \{o_{15}\}$; $\pi_c(\eta_9) = \{o_{16}\}$; $\pi_c(\eta_{10}) = \{o_{17}\}$;
$\kappa = \kappa_i \cup \kappa_o$ /* TC Assignments */	$\kappa_i(\alpha_1) = \{\emptyset\}$; $\kappa_i(\alpha_1) = \{d\}$; $\kappa_i(\alpha_2) = \{d\}$; $\kappa_i(\alpha_3) = \{d\}$; $\kappa_i(\alpha_4) = \{tc_1\}$; $\kappa_i(\alpha_5) = \{d\}$; $\kappa_i(\alpha_6) = \{tc_2\}$; $\kappa_i(\alpha_7) = \{d\}$; $\kappa_i(\alpha_8) = \{d\}$; $\kappa_i(\alpha_9) = \{tc_3\}$; $\kappa_i(\alpha_{10}) = \{tc_3\}$; $\kappa_i(\alpha_{11}) = \{tc_3\}$; $\kappa_i(\alpha_{12}) = \{tc_3\}$; $\kappa_i(\alpha_{13}) = \{d\}$; $\kappa_i(\alpha_{14}) = \{d\}$; $\kappa_i(\alpha_{15}) = \{d\}$; $\kappa_i(\alpha_{16}) = \{d\}$; $\kappa_i(\alpha_{17}) = \{tc_4\}$; $\kappa_i(\alpha_{18}) = \{d\}$; $\kappa_i(\alpha_{19}) = \{d\}$; $\kappa_i(\alpha_{20}) = \{d\}$; $\kappa_i(\alpha_{21}) = \{d\}$; $\kappa_i(\alpha_{22}) = \{tc_4\}$; $\kappa_i(\alpha_{23}) = \{d, tc_3\}$; $\kappa_i(\alpha_{24}) = \{d\}$; $\kappa_i(\alpha_{25}) = \{d, tc_3\}$; $\kappa_i(\alpha_{26}) = \{d\}$; $\kappa_i(\alpha_{27}) = \{d, tc_3\}$; $\kappa_i(\alpha_{28}) = \{d\}$; $\kappa_i(\alpha_{29}) = \{d, tc_3\}$; $\kappa_i(\alpha_{30}) = \{d\}$; $\kappa_i(\alpha_F) = \{d\}$;	$\kappa_o(\alpha_1) = \{d\}$; $\kappa_o(\alpha_1) = \{d\}$; $\kappa_o(\alpha_2) = \{d\}$; $\kappa_o(\alpha_3) = \{d\}$; $\kappa_o(\alpha_4) = \{d\}$; $\kappa_o(\alpha_5) = \{d\}$; $\kappa_o(\alpha_6) = \{d\}$; $\kappa_o(\alpha_7) = \{d\}$; $\kappa_o(\alpha_8) = \{d\}$; $\kappa_o(\alpha_9) = \{d\}$; $\kappa_o(\alpha_{10}) = \{d\}$; $\kappa_o(\alpha_{11}) = \{d\}$; $\kappa_o(\alpha_{12}) = \{d\}$; $\kappa_o(\alpha_{13}) = \{d\}$; $\kappa_o(\alpha_{14}) = \{d\}$; $\kappa_o(\alpha_{15}) = \{d\}$; $\kappa_o(\alpha_{16}) = \{d\}$; $\kappa_o(\alpha_{17}) = \{d\}$; $\kappa_o(\alpha_{18}) = \{d\}$; $\kappa_o(\alpha_{19}) = \{tc_1, tc_2\}$; $\kappa_o(\alpha_{20}) = \{d\}$; $\kappa_o(\alpha_{21}) = \{d\}$; $\kappa_o(\alpha_{22}) = \{d\}$; $\kappa_o(\alpha_{23}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_{24}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_{25}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_{26}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_{27}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_{28}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_{29}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_{30}) = \{tc_3, tc_4\}$; $\kappa_o(\alpha_F) = \{\emptyset\}$;

on the organization's employment policy. If the results satisfy the policy, then the actor prepares and informs so that the clerks can proceed continuously to the internal hiring procedure.

- The DATABASE UPDATE activity-Hiring (α_{15}) updates the employment database automatically.

Actors and Roles

There are ten roles—applicant, hiring clerk, hiring manager, personnel clerk, personnel manager, medical clerk, medical manager, employment clerk, employment manager and computer—and seventeen actors in the hiring workflow procedure. The basic principle of role-actor association is many-to-many association; an actor may be involved in several roles at the same time and *vice versa*. The left-hand side of Figure 5 presents the ICN-based hiring workflow model and its role and actor assignments. In the role assignments function($\varepsilon = \varepsilon_p \cup \varepsilon_a$), as you can recognize that the role of hiring clerk(η_{15}) has 6 subgroups ($\eta_2 = \{\eta_{2,1} \cup \eta_{2,2} \cup \eta_{2,3} \cup \eta_{2,4} \cup \eta_{2,5} \cup \eta_{2,5}\}$) it is possible for a role to be made up of several subgroups.

Relevant Data

There are typically five relevant data within the hiring workflow model: application information, decision result, checkup done, checkup results, and review results. In fact, there are other relevant data for processing applications, but we do not specify the details here to simplify the model. The right-hand side of Figure 4 depicts the relevant data flows and assignments (access mode : read or write) on each of the activities.

According to the formal definition of the ICN-based workflow model, we try to graphically define the hiring workflow procedure as shown in Figure 4 and Figure 5. Figure 4 shows the control flow (temporal orders) and data flow (input/output relevant data on repository) among the activities in the hiring workflow procedure, and Figure 5 graphically presents the activity-role association and the role-actor association in the hiring workflow procedure. Based upon the graphical definition of the ICN-based workflow model, we also give a formal representation of the hiring workflow procedure as shown in Table 1 and Table 2, which is made up by the execution results of the functional components, such as δ, ρ, λ, ε, π where δ, ρ, λ, ε, π and κ represent control flows, data flows, invoked application program associations, role associations, actor associations and transition condition associations, respectively.

ADVANCED WORKFLOW MODELS

In the previous section, we defined the ICN-based workflow model and showed how it works through graphical and formal representations of the hiring workflow procedure as an example. Once we define a workflow procedure by an ICN-based workflow modeling tool, the defined model is interpreted into one of the standard forms of process definition languages such as WPDL[6] and XPDL[7], and eventually it will be stored onto a database organized by a workflow process schema based on the workflow meta-model. From the database, we are able to derive various sorts of workflow-related knowledge and information. These derived knowledge and information can be effectively used for embodying advanced workflow architectures which are sophisticatedly implementable for the advanced computing environments like a grid computing environment, and the special domains of workflow applications, such as collaborative and scientific workflow applications. Therefore, in this section, we introduce two advanced workflow models—role-based model and actor-based model—that are automatically derived from the original ICN-based workflow model, and we show that these two advanced models are fitted very well into organizing role-based workflow

architecture as well as actor-based workflow architecture, respectively.

Role-Based Workflow Model

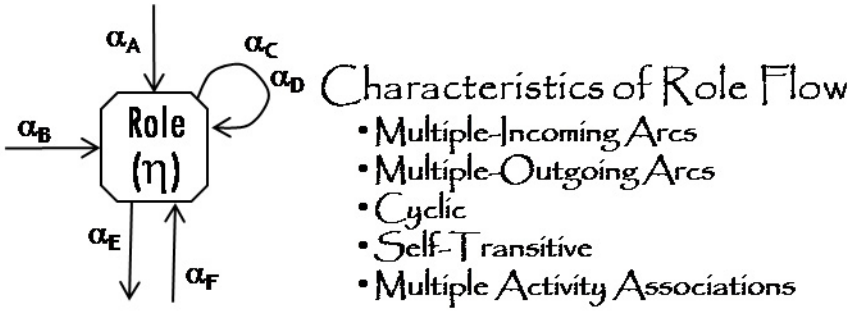
In this subsection, we define the basic concept of role-based workflow model and its graphical and formal representations. Basically, the primary goal of role-based workflow model is to make a reasonable workflow model to be applied into a workflow architecture that is appropriate to the recent working behaviors as well as the newly emerging computing environments. In terms of the working behaviors, the traditional workflow management systems support the so-called *uni-casting* work item delivery because of delivering a work item to an exact actor through pushing mechanism of the workflow engine, while on the other the recent working behaviors require the so-called *any-casting* work item delivery as well as the *multi-casting* work item delivery, which mean that not only anyone out of a group of actors is able to voluntarily pull a work item, but also all actors of the group of actors are able to collaboratively work on a single work item at the same time. The role-based workflow model is able to efficiently describe these working behaviors. Also, the grid computing environment is one of the newly emerging computing environments, and the role-based workflow model can be a reasonable theoretical basis for workflow architectures deployed over a grid computing environment.

The formal definition of the role-based workflow model is described in [**Definition 2**], and its graphical primitive is illustrated in Figure 6. The model represents two types of information— role flows and acquisition activities of roles— through which we are able to get precedence (predecessor/successor) relationships among roles as well as acquired activities of each role in a workflow procedure. The activities on the incoming directed arcs, such as $\alpha_A, \alpha_B, \alpha_C, \alpha_D, \alpha_F$, are the previously performed activities by the predecessor of the role, η, and the activities on the outgoing directed arcs, such as $\alpha_E, \alpha_C, \alpha_D$, are the acquisition activities of the role, η. And besides, the activities, α_C, α_D, on the transitive directed arc imply not only the acquired activities of the role but also the previous activities of the role, itself. As stated in the figure, the characteristics of the role flow graph are multiple-incoming arcs, multiple-outgoing arcs, cyclic, self-transitive, and multiple-activity associations on arc.

Definition 2. Role-based Workflow Model *A role-based workflow model is formally defined as* $\mathfrak{R} = (\xi, \vartheta, \mathbf{S}, \mathbf{E})$, *over a set* **P** *of roles and a set* **A** *of activities, where,*

- **S** *is a finite set of the initial roles connected from some external role-based workflow models;*

Figure 6. The Primitive of Role-based Workflow Model



- **E** *is a finite set of the final roles connected to some external role-based workflow models.*
- $\xi = \xi_i \cup \xi_o$ */* Role Flow: successors and predecessors */* *where* $\xi_o : \mathbf{P} \to \wp(\mathbf{P})$*, is a multi-valued function mapping a role to its sets of (immediate) successors, and* $\xi_i : \mathbf{P} \to \wp(\mathbf{P})$ *is a multi-valued function mapping a role to its sets of (immediate) predecessors;*
- $\vartheta = \vartheta_i \cup \vartheta_o$ */* previous worked and acquisition activities */* *where,* $\vartheta_i : \mathbf{P} \to \wp(\mathbf{P})$ *is a multi-valued function returning a set of previously worked activities,* $\mathbf{J} \subseteq \mathbf{A}$ *, on directed arcs,* $(\xi_i(\eta), \eta), \eta \in \mathbf{P}$*, from* $\xi_i(\eta)$ *to* η*; and* $\vartheta_o : \mathbf{P} \to \wp(\mathbf{P})$ *is a multi-valued function returning a set of acquisition activities,* $\mathbf{J} \subseteq \mathbf{A}$ *, on directed arcs,* $(\eta, \xi_o(\eta)), \eta \in \mathbf{P}$*, from* η *to* $\xi_o(\eta)$*;*

In terms of designing and modeling a workflow procedure, it is definitely inconvenient for us to design the workflow procedure by using the role-based workflow model. In other words, it is very important to provide a modeling methodology based upon the conventional workflow modeling approaches. Therefore, we conceive an automatic modeling methodology for the role-based workflow model, which algorithmically constructs a role-based workflow model from an ICN-based workflow model. The following is the algorithm automatically extracting a role-based workflow model from an ICN-based workflow model.

The Construction Algorithm for Role-based Workflow Model. A sketch of the algorithm is given as the following:

Input An ICN, $\Gamma = (\delta, \rho, \lambda, \varepsilon, \pi, \kappa, \mathbf{I}, \mathbf{O})$;
Output A Role-based Workflow Model, $\mathfrak{R} = (\xi, \vartheta, \mathbf{S}, \mathbf{E})$;
Begin Procedure
For $(\forall \alpha \in A)$ **Do**
Begin
/* $\xi = \xi_i \cup \xi_o$ */

Figure 7. A Role-based Model of the Hiring Workflow Procedure

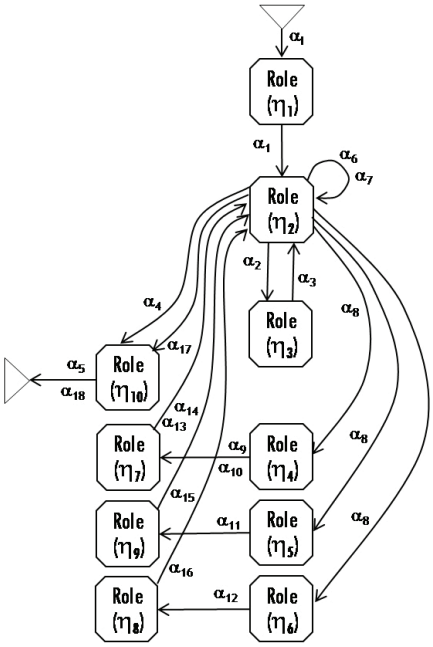


Table 3. Formal Representation of the Role-based Hiring Workflow Model

$\Re = (\xi, \vartheta, \mathbf{S}, \mathbf{E})$ over $\mathbf{A}, \mathbf{P}$	The Role-based Hiring Workflow Model	
$\mathbf{A} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8, \alpha_9, \alpha_{10}, \alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_{14}, \alpha_{15}, \alpha_{16}, \alpha_{17}, \alpha_{18}, \alpha_I, \alpha_F\}$	Elementary Activities	
$\mathbf{P} = \{\eta_I, \eta_1, \eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_7, \eta_8, \eta_9, \eta_{10}, \eta_F\}$ $\eta_2 = \{\eta_{2.1} \cup \eta_{2.2} \cup \eta_{2.3} \cup \eta_{2.4} \cup \eta_{2.5} \cup \eta_{2.6}\}$	Roles	
$\mathbf{S} = \{\emptyset\}$	Initial Roles from some external role-based workflow models	
$\mathbf{E} = \{\emptyset\}$	Final Roles to some external role-based workflow models	
$\xi = \xi_i \cup \xi_o$	ξ_i:**Predecessors**	ξ_o:**Successors**
	$\xi_i(\eta_I) = \{\emptyset\};$	$\xi_o(\eta_I) = \{\eta_1\};$
	$\xi_i(\eta_1) = \{\eta_I\};$	$\xi_o(\eta_1) = \{\eta_2\};$
	$\xi_i(\eta_2) = \{\eta_1, \eta_2, \eta_7, \eta_8, \eta_9\};$	$\xi_o(\eta_2) = \{\eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_{10}\};$
	$\xi_i(\eta_3) = \{\eta_2\};$	$\xi_o(\eta_3) = \{\eta_2\};$
	$\xi_i(\eta_4) = \{\eta_2\};$	$\xi_o(\eta_4) = \{\eta_7\};$
	$\xi_i(\eta_5) = \{\eta_2\};$	$\xi_o(\eta_5) = \{\eta_9\};$
	$\xi_i(\eta_6) = \{\eta_2\};$	$\xi_o(\eta_6) = \{\eta_8\};$
	$\xi_i(\eta_7) = \{\eta_4\};$	$\xi_o(\eta_7) = \{\eta_2\};$
	$\xi_i(\eta_8) = \{\eta_6\};$	$\xi_o(\eta_8) = \{\eta_2\};$
	$\xi_i(\eta_9) = \{\eta_5\};$	$\xi_o(\eta_9) = \{\eta_2\};$
	$\xi_i(\eta_{10}) = \{\eta_2\};$	$\xi_o(\eta_{10}) = \{\eta_F\};$
	$\xi_i(\eta_F) = \{\eta_7\};$	$\xi_o(\eta_F) = \{\emptyset\};$
$\vartheta = \vartheta_i \cup \vartheta_o$	ϑ_i:**PreviousWorked**	ϑ_o:**AcquisitionWork**
	$\vartheta_i(\eta_I) = \{\emptyset\};$	$\vartheta_o(\eta_I) = \{(\alpha_I, \eta_1)\};$
	$\vartheta_i(\eta_1) = \{(\alpha_I, \eta_I)\};$	$\vartheta_o(\eta_1) = \{(\alpha_1, \eta_2)\};$
	$\vartheta_i(\eta_2) = \{(\alpha_1, \eta_1), (\alpha_3, \eta_3), (\alpha_6, \eta_2), (\alpha_7, \eta_2), (\alpha_{13}, \eta_7), (\alpha_{14}, \eta_7), (\alpha_{15}, \eta_9), (\alpha_{16}, \eta_8)\};$	$\vartheta_o(\eta_2) = \{(\alpha_2, \eta_3), (\alpha_4, \eta_{10}), (\alpha_6, \eta_2), (\alpha_7, \eta_2), (\alpha_8, \eta_4), (\alpha_8, \eta_5), (\alpha_8, \eta_6), (\alpha_{17}, \eta_{10})\};$
	$\vartheta_i(\eta_3) = \{(\alpha_2, \eta_2)\};$	$\vartheta_o(\eta_3) = \{(\alpha_3, \eta_2)\};$
	$\vartheta_i(\eta_4) = \{(\alpha_8, \eta_2)\};$	$\vartheta_o(\eta_4) = \{(\alpha_9, \eta_7), (\alpha_{10}, \eta_7)\};$
	$\vartheta_i(\eta_5) = \{(\alpha_8, \eta_2)\};$	$\vartheta_o(\eta_5) = \{(\alpha_{11}, \eta_9)\};$
	$\vartheta_i(\eta_6) = \{(\alpha_8, \eta_2)\};$	$\vartheta_o(\eta_6) = \{(\alpha_{12}, \eta_8)\};$
	$\vartheta_i(\eta_7) = \{(\alpha_9, \eta_4), (\alpha_{10}, \eta_4)\};$	$\vartheta_o(\eta_7) = \{(\alpha_{13}, \eta_2), (\alpha_{14}, \eta_2)\};$
	$\vartheta_i(\eta_8) = \{(\alpha_{12}, \eta_6)\};$	$\vartheta_o(\eta_8) = \{(\alpha_{16}, \eta_2)\};$
	$\vartheta_i(\eta_9) = \{(\alpha_{11}, \eta_5)\};$	$\vartheta_o(\eta_9) = \{(\alpha_{15}, \eta_2)\};$
	$\vartheta_i(\eta_{10}) = \{(\alpha_4, \eta_2), (\alpha_{17}, \eta_2)\};$	$\vartheta_o(\eta_{10}) = \{(\alpha_5, \eta_F), (\alpha_{18}, \eta_F)\};$
	$\vartheta_i(\eta_F) = \{(\alpha_5, \eta_{10}), (\alpha_{18}, \eta_{10})\};$	$\vartheta_o(\eta_F) = \{\emptyset\};$

Add *all members of* $\varepsilon_p(\alpha)$ **To** $\xi_i(\varepsilon_p(\delta_o(\alpha)))$;
Add *all members of* $\varepsilon_p(\delta_o(\alpha))$ **To** $\xi_i(\varepsilon_p(\alpha))$;
/* $\vartheta = \vartheta_i \cup \vartheta_o$ */
Add *arc* $(\alpha, \varepsilon_p(\alpha))$ **To** $\vartheta_i(\varepsilon_p(\delta_o(\alpha)))$
Add *arc* $(\alpha, \varepsilon_p(\delta_o(\alpha)))$ **To** $\vartheta_i(\varepsilon_p(\alpha))$
End
End Procedure

As a result, we show a role-based workflow model of the hiring workflow procedure in Figure 7, which is automatically generated by applying the algorithm to the ICN-based hiring workflow model. As you can see, the role-based hiring workflow model shows the role flow information and each role's acquition activities based upon 10 roles and 18 elementary activities. Note that we did not handle the subgroups $(\eta_{2.1}, \eta_{2.2}, \eta_{2.3}, \eta_{2.4}, \eta_{2.5}, \eta_{2.6})$ of the role, η_2, in this case, for the sake of simplification. However, without any revisions of the algorithm, the role-based hiring workflow model can be modified if the subgroups are taken into the algorithm as input. Also the algorithm ignores the gateway activities including block activities, because role assignments have nothing to do with these special types of activities. Accordingly, Table. 3 is the final outcomes of execution of the algorithm by inputting the ICN-based hiring workflow model, and finally we summarize them as the formal representation of the role-based hiring workflow model.

Actor-Based Workflow Model

In this subsection, we introduce the basic concept and definition of actor-based workflow model as

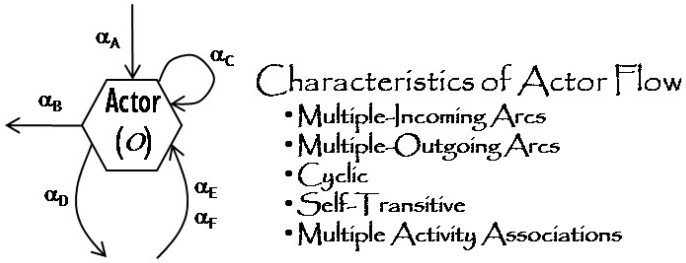
the second one out of the advanced workflow models. The conceptual background of the actor-based workflow model comes from the fact that it should be able to incorporate the advanced technological and organizational trends into workflow model in order for the underlying workflow management system to effectively acclimate itself to a new technological and organizational environment. That is, the recent technological trends in workflow area can be characterized by increasingly powerful networked personal computing facilities (P2P[8] computing environments) and increasingly large and complex workflow applications. And, if the concept of actor orientation can be embedded into a workflow model, then the workflow model ought to be more effective and efficient in these evolving technological trends. Therefore, we define an actor-based workflow model so as to effectively model and design coordination among humans (actors) who handle activities associated in a workflow procedure. This subsection addresses the actor-based workflow model by systematically and formally formulating a way to describe and incorporate the concept of actor orientation into workflow models and architectures.

Basically, the actor-based workflow model represents the behaviors of acquisitioning activities among actors associated in a workflow procedure. The formal definition of the actor-based workflow model is given in [**Definition 3**], and its graphical primitive construct is illustrated in Figure 8. The behaviors of the model are revealed through incoming and outgoing directed arcs labeled with activities associated with each of actors. The directed arcs imply two kinds of behaviors—actor flows and acquisition activities of actors—through which we are able to get precedence (candidate-predecessor/candidate-successor) relationships among actors as well as acquisition activities of each actor in a workflow procedure. In terms of defining actor's predecessors and successors, we would use the prepositional word, "**candidate**", because, unlike in the role-based workflow model, in actor-based workflow model a role-actor mapping is an one-to-many relationship, and the actor selection mechanism will choose one actor out of the assigned actors mapped to the corresponding role during the workflow model's runtime.

The activities on the incoming directed arcs, such as $\alpha_A, \alpha_C, \alpha_E, \alpha_F$, are the previously performed activities by the predecessors of the actor, *O*, and the activities on the outgoing directed arcs, such as $\alpha_B, \alpha_C, \alpha_D$, are the activities acquired by the actor, itself. And besides, the activity, α_C, on the transitive directed arc implies not only the acquisitioning activities of the actor but also the previously performed activities by the actor, itself. As stated in the figure, the characteristics of the actor flow graph are multiple-incoming arcs, multiple-outgoing arcs, cyclic, self-transitive, and multiple-activity associations on arcs.

Definition 3. Actor-based Workflow Model. *An actor-based workflow model is formally defined as* $\Lambda = (\sigma, \psi, \mathbf{S}, \mathbf{E})$, *over a set* **C** *of actors, and a set* **A** *of activities, where*

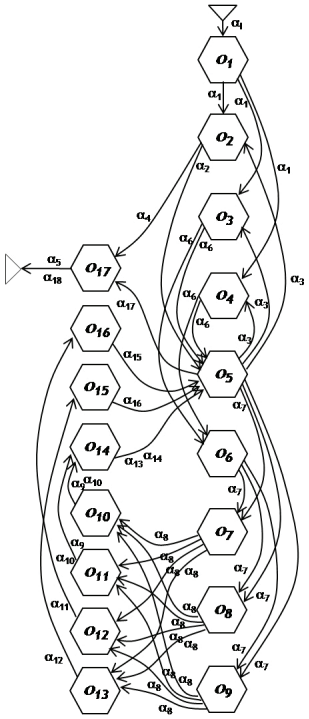
Figure 8. The Hiring of Actor-based Workflow Model



- **S** *is a finite set of coordinators or coordinator-groups connected from some external actor-based workflow models;*
- **E** *is a finite set of coordinators or coordinator-groups connected to some external actor-based workflow models;*
- $\sigma = \sigma_i \cup \sigma_o$ */* Actor Flow: successors and predecessors */* *where,* $\sigma_o : \mathbf{P} \rightarrow \wp(\mathbf{P})$ *is a multi-valued function mapping an actor to its sets of (immediate) candidate-successors, and* $\sigma_i : \mathbf{P} \rightarrow \wp(\mathbf{P})$ *is a multi-valued function mapping an actor to its sets of (immediate) candidate-predecessors;*
- $\psi = \psi \cup \psi_o$ */* acquisition activities */* *where,* $\psi_i : \mathbf{P} \rightarrow \wp(\mathbf{P})$ *is a multi-valued function returning a bag of previously worked activities,* $(\mathbf{K} \subseteq \mathbf{A})$*, on directed arcs,* $(\sigma_i(o), o), o \in C$ *, from* $\sigma_i(o)$ *to o; and* $\psi_o : \mathbf{P} \rightarrow \wp(\mathbf{P})$ *is a multi-valued function returning a set of acquisition activities,* $(\mathbf{K} \subseteq \mathbf{A})$ *, on directed arcs,* $(o, \sigma_o(o)), o \in C$ *from o to* $\sigma_o(o)$*;*

Likewise, the actor-based workflow modeling methodology might not be a convenient work in terms of designing and modeling a workflow procedure, too. In other words, it is very important for the actor-based workflow model to provide an effective modeling tool with keeping the conventional workflow modeling approaches. Therefore, we conceive an automatic modeling methodology for the actor-based workflow model, which algorithmically constructs an actor-based workflow model from an ICN-based workflow model. The following is the construction algorithm for automatically extracting an actor-based workflow model from an ICN-based workflow model. Particularly, in order to construct an actor-based workflow model, it needs, as inputs, the sets of δ_O (control flow information), ε_P (activity-role mapping information) and π_C (role-actor mapping

Figure 9. An Actor-based Model of the Hiring Workflow Procedure



information) in an ICN-based workflow model. Additionally, we have to remind that a group of actors can cooperatively and simultaneously perform an activity, which is called a realtime groupware activity, and almost all current available workflow models do not support such realtime groupware activities. Therefore, the actor-based workflow model is able to provide a feasible solution for supporting such realtime groupware activities, which is one of the cutting-edge workflow features to be required in the future-generation workflow management systems.

The Construction Algorithm for the Actor-based Workflow Model. An actor-based workflow model is constructed from an ICN-based workflow model through the following algorithm:

Input An ICN, $\Gamma = (\delta, \rho, \lambda, \varepsilon, \pi, \kappa, \mathbf{I}, \mathbf{O})$;
Output A Actor-based Workflow Model, $\Lambda = (\sigma, \psi, \mathbf{S}, \mathbf{E})$;
Begin Procedure
For $(\forall \alpha \in A)$ **Do**
Begin
/* $\sigma = \sigma_i \cup \sigma_o$ */
Add *all members of* $\pi_C(\varepsilon_p(\alpha))$ **To**
σ_i(*each member of* $[\pi_C(\varepsilon)]_p(\delta_o(\alpha))$);

Table 4. Formal Representation of the Actor-based Hiring Workflow Model

$\Lambda = (\sigma, \psi, \mathbf{S}, \mathbf{E})$ over $\mathbf{A}, \mathbf{C}$ — The Actor-based Hiring Workflow Model

$\mathbf{A} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_8, \alpha_9, \alpha_{10}, \alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_{14}, \alpha_{15}, \alpha_{16}, \alpha_{17}, \alpha_{18}, \alpha_I, \alpha_F\}$ Elementary Activities
$\mathbf{C} = \{o_I, o_1, o_2, o_3, o_4, o_5, o_6, o_7\ o_8, o_9, o_{10}, o_{11}, o_{12}, o_{13}, o_{14}, o_{15}, o_{16}, o_{17}, o_F\}$ Actors
$\mathbf{S} = \{\emptyset\}$ Initial Actors from some external actor-based workflow models
$\mathbf{E} = \{\emptyset\}$ Final Actors to some external actor-based workflow models

$\sigma = \sigma_i \cup \sigma_o$

σ_i:**CandidatePredecessors**

$\sigma_i(o_I) = \{\emptyset\}$;
$\sigma_i(o_1) = \{o_I\}$;
$\sigma_i(o_2) = \{o_2, o_5\}$;
$\sigma_i(o_3) = \{o_1, o_5\}$;
$\sigma_i(o_4) = \{o_1, o_5\}$;
$\sigma_i(o_5) = \{o_2, o_3, o_4, o_{14}, o_{15}, o_{16}\}$;
$\sigma_i(o_6) = \{o_3, o_4\}$;
$\sigma_i(o_7) = \{o_5, o_6\}$;
$\sigma_i(o_8) = \{o_5, o_6\}$;
$\sigma_i(o_9) = \{o_5, o_6\}$;
$\sigma_i(o_{10}) = \{o_7, o_8, o_9\}$;
$\sigma_i(o_{11}) = \{o_7, o_8, o_9\}$;
$\sigma_i(o_{12}) = \{o_7, o_8, o_9\}$;
$\sigma_i(o_{13}) = \{o_7, o_8, o_9\}$;
$\sigma_i(o_{14}) = \{o_{10}, o_{11}\}$;
$\sigma_i(o_{15}) = \{o_{13}\}$;
$\sigma_i(o_{16}) = \{o_{12}\}$;
$\sigma_i(o_{17}) = \{o_5, o_2\}$;
$\sigma_i(o_F) = \{o_{17}\}$;

σ_o:**CandidateSuccessors**

$\sigma_o(o_I) = \{o_1\}$;
$\sigma_o(o_1) = \{o_2, o_3, o_4\}$;
$\sigma_o(o_2) = \{o_5, o_{17}\}$;
$\sigma_o(o_3) = \{o_5, o_6\}$;
$\sigma_o(o_4) = \{o_5, o_6\}$;
$\sigma_o(o_5) = \{o_2, o_3, o_4, o_7, o_8, o_9, o_{17}\}$;
$\sigma_o(o_6) = \{o_7, o_8, o_9\}$;
$\sigma_o(o_7) = \{o_{10}, o_{11}, o_{12}, o_{13}\}$;
$\sigma_o(o_8) = \{o_{10}, o_{11}, o_{12}, o_{13}\}$;
$\sigma_o(o_9) = \{o_{10}, o_{11}, o_{12}, o_{13}\}$;
$\sigma_o(o_{10}) = \{o_{14}\}$;
$\sigma_o(o_{11}) = \{o_{14}\}$;
$\sigma_o(o_{12}) = \{o_{16}\}$;
$\sigma_o(o_{13}) = \{o_{15}\}$;
$\sigma_o(o_{14}) = \{o_5\}$;
$\sigma_o(o_{15}) = \{o_5\}$;
$\sigma_o(o_{16}) = \{o_5\}$;
$\sigma_o(o_{17}) = \{o_F\}$;
$\sigma_o(o_F) = \{\emptyset\}$;

$\psi = \psi_i \cup \psi_o$

ψ_i:**PreviousWork**

$\psi_i(o_I) = \{\emptyset\}$;
$\psi_i(o_1) = \{(\alpha_I, o_I)\}$;
$\psi_i(o_2) = \{(\alpha_1, o_1), (\alpha_3, o_5)\}$;
$\psi_i(o_3) = \{(\alpha_1, o_1, (\alpha_3, o_5)\}$;
$\psi_i(o_4) = \{(\alpha_1, o_1, (\alpha_3, o_5)\}$;
$\psi_i(o_5) = \{(\alpha_2, o_2), (\alpha_6, o_3), (\alpha_6, o_4),$
$(\alpha_{13}, o_{14}), (\alpha_{14}, o_{14}), (\alpha_{15}, o_{16}), (\alpha_{16}, o_{15})\}$;
$\psi_i(o_6) = \{(\alpha_6, o_3), (\alpha_6, o_4)\}$;
$\psi_i(o_7) = \{(\alpha_7, o_5), (\alpha_7, o_6)\}$;
$\psi_i(o_8) = \{(\alpha_7, o_5), (\alpha_7, o_6)\}$;
$\psi_i(o_9) = \{(\alpha_7, o_5), (\alpha_7, o_6)\}$;
$\psi_i(o_{10}) = \{(\alpha_8, o_7), (\alpha_8, o_8), (\alpha_8, o_9)\}$;
$\psi_i(o_{11}) = \{(\alpha_8, o_7), (\alpha_8, o_8), (\alpha_8, o_9)\}$;
$\psi_i(o_{12}) = \{(\alpha_8, o_7), (\alpha_8, o_8), (\alpha_8, o_9)\}$;
$\psi_i(o_{13}) = \{(\alpha_8, o_7), (\alpha_8, o_8), (\alpha_8, o_9)\}$;
$\psi_i(o_{14}) = \{(\alpha_9, o_{10}), (\alpha_{10}, o_{10}), (\alpha_9, o_{11}),$
$(\alpha_{10}, o_{11})\}$;
$\psi_i(o_{15}) = \{(\alpha_{12}, o_{13})\}$;
$\psi_i(o_{16}) = \{(\alpha_{11}, o_{12}\}$;
$\psi_i(o_{17}) = \{(\alpha_4, o_2), (\alpha_{17}, o_5)\}$;
$\psi_i(o_F) = \{(\alpha_5, o_{17}), (\alpha_{18}, o_{17})\}$;

ψ_o:**AcquisitionWork**

$\psi_o(o_I) = \{(\alpha_I, o_1)\}$;
$\psi_o(o_1) = \{(\alpha_1, o_2), (\alpha_1, o_3), (\alpha_1, o_4)\}$;
$\psi_o(o_2) = \{(\alpha_2, o_5), (\alpha_4, o_{17})\}$;
$\psi_o(o_3) = \{(\alpha_6, o_5), (\alpha_6, o_6)\}$;
$\psi_o(o_4) = \{(\alpha_6, o_5), (\alpha_6, o_6)\}$;
$\psi_o(o_5) = \{(\alpha_3, o_2), (\alpha_3, o_3), (\alpha_3, o_4),$
$(\alpha_7, o_7), (\alpha_7, o_8), (\alpha_7, o_9), (\alpha_{17}, o_{17})\}$;
$\psi_o(o_6) = \{(\alpha_7, o_7), (\alpha_7, o_8), (\alpha_7, o_9)\}$;
$\psi_o(o_7) = \{(\alpha_8, o_{10}), (\alpha_8, o_{11}), (\alpha_8, o_{12}),$
$(\alpha_8, o_{13})\}$;
$\psi_o(o_8) = \{(\alpha_8, o_{10}), (\alpha_8, o_{11}), (\alpha_8, o_{12}),$
$(\alpha_8, o_{13})\}$;
$\psi_o(o_9) = \{(\alpha_8, o_{10}), (\alpha_8, o_{11}), (\alpha_8, o_{12}),$
$(\alpha_8, o_{13})\}$;
$\psi_o(o_{10}) = \{(\alpha_9, o_{14}), (\alpha 10, o_{14})\}$;
$\psi_o(o_{11}) = \{(\alpha_9, o_{14}), (\alpha 10, o_{14})\}$;
$\psi_o(o_{12}) = \{(\alpha_{11}, o_{16})\}$;
$\psi_o(o_{13}) = \{(\alpha_{12}, o_{15})\}$;
$\psi_o(o_{14}) = \{(\alpha_{13}, o_5), (\alpha_{14}, o_5)\}$;
$\psi_o(o_{15}) = \{(\alpha_{16}, o_5)\}$;
$\psi_o(o_{16}) = \{(\alpha_{15}, o_5)\}$;
$\psi_o(o_{17}) = \{(\alpha_5, o_F), (\alpha_{18}, o_F)\}$;
$\psi_o(o_F) = \{\emptyset\}$;

Add *all members of* $[\![\pi_{\downarrow}C\ (\varepsilon)]\!]_{\downarrow}p\ (\delta_{\downarrow}o\ (\alpha))))(\delta_{\downarrow}o\ (\alpha))$ **To** $\sigma_o(each\ member\ of\ [\![\pi_C(\varepsilon)]\!]_p(\alpha))$;

/* $\psi = \psi_i \cup \psi_o = \psi_i \cup \psi_o$ */

Add *all pairs of* $(\alpha, o), \forall o \in \pi_C(\varepsilon(\alpha))$ **T** $\psi_{\downarrow}i$ *each member of* $\pi_{\downarrow}C\ (\varepsilon_{\downarrow}p\ (\delta_{\downarrow}o\ (\alpha))))$;

Add *all pairs of* $(\alpha, o), \forall o \in \pi_C(\varepsilon_p(\delta_O(\alpha)))$ **To** $\psi_{\downarrow}O$ *each member of* $\pi_{\downarrow}C\ (\varepsilon_{\downarrow}p\ (\alpha)))$;

End

End Procedure

As an example, we apply the algorithm to the hiring workflow procedure; the input of the algorithm is the information sets of the ICN-based hiring workflow model, and its output is the actor-based hiring workflow model graphically represented in Figure 9 as well as formally specified in Table 4. Unlike the role-based workflow model in which an activity is mapped to just a single role (one-to-one relationship), the actor-based workflow model has one-to-many relationships in the mappings of activities and actors. Because of the one-to-many relationships between activities and actors, an actor node may have several outgoing directed arcs that have the same activity as their labels. Therefore, during the runtime of an actor-based workflow model, each actor node having the same activity labeled on outgoing arcs needs to have a selection mechanism choosing one of the neighbor actors that have the same activity labeled on arcs so as to assign the activity to the chosen actor. On Figure 9, for example, the actor node, o_1, has three outgoing directed arcs labeled with the same activity, α_2, and so o_1 has to select one of the neighbor actors, o_2, o_3, o_4, so as to proceed to the selected actor after performing α_2 during runtime. There should be several actor selection mechanisms, such as random, sequential, heuristic selection mechanism and so on. Of course, it is possible for each actor node in a workflow procedure to have a different selection mechanism.

IMPLICATIONS ON WORKFLOW ARCHITECTURES

Workflow Architectural Complexity

As stated in the previous sections, the goal of a workflow management system is to orchestrate the enactment of workflow procedures, which is however becoming more and more complicated work according to rapidly scaling up the complexities of workflow procedures in terms of structure, size and workload. Accordingly, the higher degree of complexities in workflow procedures requires the architectural renovations of workflow management systems. As a consequence of this atmosphere, we need to be concerned about the advanced workflow architectures based upon the advanced workflow models presented in the previous sections. At the same time, it is necessary to make some criteria, which is called the degree of workflow architectural complexity, in order to effectively characterizing them; the workflow architectures can be classified based on the degree of workflow architectural complexity consisting of three perspectives and dimensions—Workflow Engagement, Workflow Structure and Workflow Instantiation. We describe the meanings of them as the followings:

- **Workflow engagement:** This dimension of the workflow architectural complexity has to do with the sizes of workflow's physical components that are served and engaged for a workflow management system. The physical components are directly related with the components of workflow models, such as actors, roles, activities, workflow procedures, business processes, and organizations. So, the degree of workflow commitment can be decided by the sizes of these components, and it also becomes an essential criterion that is able to characterize the system's scalability and performance - How well the workflow and business process management system

handles, commits, and serves for a huge number of requests coming out of a large number of actors, procedures and organizational compartments, which are involved in one or even more organizations.

- **Workflow structure:** This dimension of the workflow architectural complexity has something to do with interactional complexities between activities within a workflow procedure and between collaborative activities that are involved in interactions between workflow procedures in collaboration. In a large scale workflow procedure, several hundreds of activities might be co-associated with each other in diverse types of complex relationships, such as subworkflow, conjunctive, disjunctive, massively parallel, and cooperative relationships. Also, the recent e-Commerce, e-Logistics and e-Government marketplaces require more complicated relationships for supporting the inter-organizational workflow services like collaborative workflows, choreographical workflows, orchestrated workflows, and chained workflows. So, these structural complexity ought to be a critical criterion for deciding the degree of workflow architectural complexity.
- **Workflow instantiation:** This dimension of the workflow architectural complexity has directly related with measurement of the system's performance. In a large scale workflow application domain, the number of workcases instantiated from a workflow procedure may be from several hundred-thousands of workcases to several millions of workcases. Also, the large number of workflow procedures can be managed and maintained in a workflow management system. Specially, this dimension is inflicting heavy effect on the fundamental infrastructures of the workflow management systems. That is, the systems can be extensively deployed on distributed computing environments, such as clustered, grid, and peer-to-peer computing architectures, in order to handle those large scale workcases without any further performance degradation. So, this dimension should be the most important criterion in deciding the degree of workflow architectural complexity.

The workflow architectural renovations should be accomplished by supporting the highest levels of services in all three dimensions. Particularly, in order to be satisfied with the highest levels of services in both the workflow engagement dimension and the workflow structure dimension, it is necessary for the system to be completely renovated from the fundamental and philosophical reformations, whereas the highest level of the workflow instantiation can be renovated only through the software architectural reformations. We would expect that the role-based workflow model and the actor-based workflow model presented in this chapter ought to be the theoretical basis for the architectural renovations satisfying the higher levels of complexities in the workflow engagement dimension and the workflow structure dimension.

Workflow Architectural Framework

In order to systematically support the workflow architectural renovations, we conceived an architectural framework in [34]. The architectural framework suggests that there are some overriding elements that generically characterize any workflow system. These elements are captured in the generic level of the architectural description, and include elements of workflow execution, control, and data. At a more detailed level, every workflow system deals with some set of conceptual objects such as workcases, activities, actors, and roles. These objects, and their representation are dealt with in the conceptual level of the architectural description. Finally, the lowest, most detailed level of our framework is concerned with the physical

components of a workflow system. At this level we specify the details of processors, workstations, and networks. This level is concerned with implementation details of concrete elements, their functionality, and their inter-connectivity. Conclusively, the framework has the three levels of considerations—generic level, conceptual level, and implementation level.

Generic Level of Consideration. At the generic level, a common high-level framework from which to view the entire architectural style of a workflow system is defined. This architectural style is characterized by distribution choices selected for the execution component, the control component, and the data component of a workflow system. the structural categories possible for each component are centralized, decentralized, and dispersed. Note that the terms decentralized and dispersed refer to replicated and partitioned systems respectively.

- **Centralized:** Control, application data and / or execution scripts are kept at a single site. A site is a node of the workflow network.
- **Decentralized (replicated):** Full copies of control, application data, and / or execution scripts are distributed to more than one site.
- **Dispersed (partitioned):** Control, application data, and / or execution scripts are divided into multiple partitions, and the various partitions are kept, in a non-replicated fashion, at different sites.

Conceptual Level of Consideration. The conceptual level of our framework elucidates a taxonomy based upon the workflow concepts of activities, roles, actors, and workcases. At the generic level, we dealt with elements that are indeed generic (control, data, execution), and generally applicable to all distributed systems. While on the other, at the conceptual level, we will deal with concepts that are not generic, but specific to workflow systems. The choice of concepts follows the standards recommendation of the workflow management coalition (WfMC). The concepts will be introduced with respect to the control dimension of the generic architecture, but, as we shall see, the same concepts also serve to partition the possibilities for data and scripts. In result, at the conceptual level, an architectural framework needs to deal with issues of concept embodiment, relationships among concepts, concept communication, and conceptual responsibility distribution.

The notion of workflow concept embodiment is related to the recent notions of software architectures that argue that careful choices concerning implementation of active entities as processes, threads, or active agents has tremendous influence upon performance of a system. Recent literature has shown that these software choices are equally as important for performance as hardware choices. All workflow systems must keep data, or some representation of concepts such as activities and actors. A few modern architectures have recently implemented some of these concepts as active processes rather than as passive data. We argue that there are a plethora of unexplored workflow architectures that are radical in that they embody activities, roles, actors, and / or workcases as active processes. We specifically capture this notion of active versus passive in our conceptual level taxonomy.

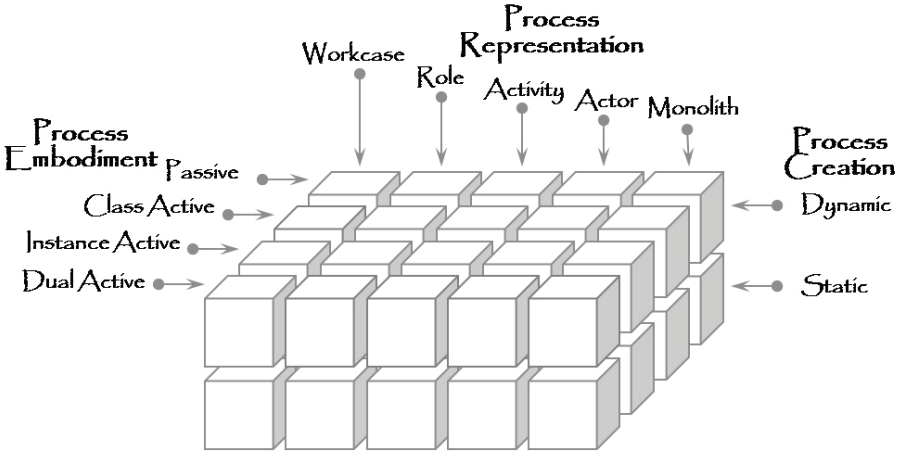
Relationships among concepts: There are numerous relationships between these workflow concepts that must be supported within a workflow architecture. The precedence relationship exists among activities; the performer relation exists between roles and activities; the part-of relationship exists between activities and procedures; etc. Some of these relationships can be specified (bound) at system creation time; others may be fixed (bound) on a per workcase basis at the time of workcase creation; still others need to be changed in various dynamic timeframes. For example, when a new application needs ad-

ministrative processing in the hiring workflow procedure, a choice must be made of which secretary will do the work. This is a binding of actor to activity that, in many environments is best done (by a workflow scheduler or a manager) as a dynamic late binding. It depends upon who is available, who is best qualified, who dealt with this application in the past, and other factors as well. In our taxonomy, we specifically capture notions of relationships, and there binding time, recognizing that there is a spectrum of binding choices ranging from static fixed, through periodic binding, opportunistic binding, up to last minute dynamic binding.

Responsibility distribution: There exist numerous options and decisions about placement and movement of data, scripts, and control information; many of these options remain unexplored in today's workflow products. Within all workflow architectures, there must be entities responsible for answering questions such as "what activity should be executed next?" and "when should this application data be moved to another site?" and "where is the most efficient site to execute this script?" Again we note that in the majority of today's workflow products, the static answer to many of these questions is "at the workflow server." In the large scale workflow systems of the near future, this answer is inadequate. There is a pressing need to explore workflow solutions that are much more distributed. One of the areas needing research is the communication between concepts once placement and movement are not simply always at the server. For example, within the JOIN operation when several activities must all finish before a successor activity can commence, how do we efficiently implement a distributed join? If the server is a bottleneck, how do we open up communication between activity entities, which can greatly decrease traffic to the server? As the size of workflow applications grows larger and larger, these issues become more pressing.

Based upon those architectural considerations, we can derive our conceptual taxonomy as shown in Figure 10, in which there are three dimensions—process embodiment, process representation and process creation—corresponding to the four relevant concepts of workcase, activity, role, and actor. We would suggest that further dimensions can and should be added in the future. In each dimension, we can choose to implement control of the concept passively as data available on the server or elsewhere, or we can implement control actively with one active agent for each instance, or with one active agent for each class, or with both (dual active). The processes that are

Figure 10. The Conceptual Properties of Workflow Architectures



created, manipulated, and destroyed in the active schemes can be static or dynamic. Note that replicated or partitioned or hybrid schemes can be used, and the same dimensions can be used for data and scripts.

Implementation Level of Consideration. The implementation level is the lowest level of the architectural framework. Components at this level include processors, networks, and memories. If there are questions concerning the performance of an architecture, then this level must be involved in the answer. Specifications of processor speeds, memory sizes, and network parameters are given at this level. In the previous levels, we were concerned with generic and conceptual entities, but not with where those entities reside in the network. Pragmatic questions of actual distribution are answered at this level. Thus, the question of which memory module contains our application data, or which processor executes a script, are implementation level details. This level can be modeled or implemented in great detail or little detail. Issues of caching, multiprocessing, threads versus process, network protocols, etc, may all be of potential concern at this level. Additionally, a complex and important consideration is the internetwork configuration. Fully connected networks have very different performance characteristics from store and forward. Broadcast media such as Ethernet are very different from point to point systems with switches. Satellite speeds for bulk transfer are very different from coaxial cable telephone line speeds.

Advanced Workflow Architectures

Based upon the conceptual taxonomy of Figure 10, we can selectively extract several advanced workflow architectures chosen from the boxes of each process embodiment dimension (concept) ─workcase, role, activity, and actor. Particularly, we remind that the role-based workflow model and the actor-based workflow model introduced in the previous sections of this chapter become the theoretical bases for the role-based workflow architecture and the actor-based workflow architecture, respectively. The followings are the details of the advanced workflow architectures that are possibly derived from the conceptual workflow taxonomy:

The active workcase dimension implies that architectures can be created in which the workcase is not simply represented as data, but as a software process that actively guides the workcase from activity to activity helping to find and select actors, roles, data, etc, for each activity. Class active, workcase centric architecture means that there is one workcase process that supervises all of the workcase instances. Tasks of this process include maintenance of all workcase histories, scheduling tasks, and up-to-date knowledge of the state of each workcase. External queries of the current status of any workcase instance are directed to this process. Instance active, workcase centric architecture means that there is one workcase process for each workcase instance. Again, this process maintains state and history, and performs other tasks. It can also act as an active negotiator for the workcase with the goal of expeditiously completing the task. It can thus help to perform activities, and perform negotiation on behalf of the workcase. This abstraction supports architectures of migrating intelligent forms.

The active activity dimension suggests an architecture in which each activity type, or each activity instance, is represented by a process. We call these architectures activity centric architectures. The five dashed ovals in figure 1 each represent the domain of a class active, activity centric process. In an instance active architecture, each of the 15 activities in figure 1 would posses its own dedicated process. Since each dimension is independent of the others, it is possible to simultaneously have software processes for all workcases, and for all activities. This implies that when a particular workcase is enabled to perform a particular activity, a cou-

pling operation is performed in which the two (or more) processes can very efficiently share data, and work together to perform the activity. Note that there may be workcase specific goals and data and scripts controlled by the workcase process; and activity specific goals and data and scripts controlled by the activity process. The coupling can be an efficient and elegant means of effectively performing the activity.

Role processes do the work of selecting which roles will be chosen to perform (or supervise, or otherwise participate in) which activities. In general this may require the evaluation of a complex network. Selection of the class active, role centric architecture implies that there is a role server process somewhere on the network. Note that it can be replicated or partitioned. This process also maintains the history of all roles, and is able to answer queries concerning all role assignments. Note that certain roles may be privileged to access certain sensitive data (e.g. management salaries or customer credit) that is not available to the workflow system in general. In this case, it can be quite useful to implement an instance active role centric architecture that features one role process for each role instance. Thus, the manager role can have access to, and protect, different data than the credit clerk role.

The actor centric architecture is similar to the role centric architecture except that, in the instance active case, there can be one for each person in the system although several people may be assigned to the same role. Actor processes do the work of selecting which actors will be chosen to play which roles, and perform which activities. In general this may require the evaluation of a complex network. These active agents maintain history and current actor status, do scheduling, and reply to relevant external queries. Selection of the class active, actor centric architecture implies that there is an actor server process somewhere on the network. At the conceptual level, we do not specify where this process resides. It could be on a dedicated actor server machine, or elsewhere, or migrating. There can also be a mixture of instance active processes and class active processes which all couple to perform a complex activity. We present an example of this later. We do not specify how the coupling is accomplished. This depends upon the network interconnection parameters, and other details, which are properly the domain of the implementation level architecture.

SUMMARY

The expressiveness of a workflow management system is determined by the content of its underlying workflow model. That is, a workflow model gives a lot of influences on design workflow procedures and implementation of a workflow management system as well. Depending on workflow model, the corresponding workflow management system may have not only different features but different methodologies. Therefore, the workflow model should incorporate the advanced technological and organizational trends so that its corresponding workflow management system enables to be effective and acclimatizes itself to a new technological and organizational working environment.

The recent trends in working environments require new types of workflow management systems that provide cooperative working facilities, by which group of people works together simultaneously, and that are equipped by increasingly large and complex workflow applications. The recent trend in technological environments in term of implementing a workflow management system is undoubtedly the object orientation. The international standard organization for object orientation, OMG (Object Management Group), has announced the workflow management facility as a standard architecture of workflow management systems. The object oriented architecture stemmed from the joint team's proposal for workflow management facility is based on activity entities. The activity based object orientation should not be

suitable for straightforwardly accommodating the recent trends in terms of the working environments supporting not only cooperative group works which are represented by the multi-cast workflow, but also pull-based works that are represented by the any-cast workflow. We strongly believe that role based workflow model is the best solution for realizing the multi-cast workflow as well as the any-cast workflow. Based on the role-based model, we can derive a design methodology for prescribing role oriented workflow procedures, and at the same time, a role oriented workflow management system. This chapter specifically addresses the Information Control Net (ICN). The ICN can be used systematically and formally to formulate a way to describe and incorporate the concept of role orientation into workflow models and architectures.

In order for a workflow process to be deployed through a workflow enactment engine, it is eventually represented by a set of activity precedence data (representing control flows), their relevant data (representing data flow), and others including their roles, actors and invoked applications, which are standardized as WPDL (Workflow Process Definition Language) or XPDL (XML Process Definition Language) by WfMC. And then the workflow enactment engine uses the activity-driven information to enact the workflow process. This is quite normal and reasonable under the traditional computing environment, because it is reflecting the way of office works in the real-world. So we call it activity-driven workflow framework. However, this should not be true if the activity-driven workflow framework without any further modification is applied into the Grid or P2P computing environment, because the Grid/P2P's resource configuration is quite inapt to the scheme of the activity-driven workflow framework. Under the Grid/P2P, the workflow process's data has to be disseminated into each of workstations and PCs—Peers. But each of the peers is owned by each of actors involved in the workflow process. Therefore, we need to reorganize the workflow process's data from the activity-driven information to actor-driven information. In other words, the actor-based workflow model ought to be the impeccable theory for accomplishing a high-level of efficiency in enacting workflows over Grid/P2P computing environment.

REFERENCES

Ferraiolo, D. F., & Kuhn, D. R. (1992). Role-based access controls. *Proceedings of the 15th NIST-NSA National Computer Security Conference*.

Ferraiolo, D. F., et al. (1995). *An introduction to role-based access control*. NIST/ITL Bulletin.

Ferraiolo, D. F., Cugini, J. A., & Kuhn, D. R. (1995). Role-based access control: Features and Motivations. *Proceedings of the 11th Annual Computer Security Applications*.

Ellis, C. A., & Nutt, G. J. (1980). Office information systems and computer science. *ACM Computing Surveys, 12*(1).

Ellis, C. A., & Nutt, G. J. (1993). The modeling and analysis of coordination systems. *University of Colorado/Dept. of Computer Science Technical Report, CU-CS-639-93*.

Ellis, C. A. (1983). Formal and informal models of office activity. *Proceedings of the 1983 Would Computer Congress*.

Bair, J. H. (1990). Contrasting workflow models: getting to the roots of three vendors. *Proceedings of International CSCW Conference*.

Kim, K., et al. (1996). Practical experience on workflow: hiring process automation by FlowMark. *IBM Internship Report, IBM/ISSC Boulder Colorado*.

Kim, K., & Paik, S. (1996). Practical experiences and requirements on workflow. *Lecture Notes Asian '96 Post-Conference Workshop: Coordina-*

tion Technology for Collaborative Applications, the 2nd Asian Computer Science Conference.

Park, M., & Kim, K. (2008). Control-path oriented workflow intelligence analyses. *Journal of information science and engineering, 34*(3).

Kim, K., & Ra, I. (2007). e-Lollapalooza: a process-driven e-Business service integration system for e-Logistics services. *KSII Transactions on Internet and Information Systems, 1*(1), 33-52.

Kim, K. (2007). Signal-Algorithm: structured workflow process mining through amalgamating temporal workcases. *Lecture Notes in Artificial Intelligence, 4426,* 119-130.

Kim, K. (2007). A layered workflow knowledge Grid/P2P architecture and its models for future generation workflow systems. *Future Generation Computer Systems, 23*(3), 304-316.

Kim, K. (2006). Beyond Workflow Mining. *Lecture Notes in Computer Science, 4102,* 49-64.

Kim, K. (2006). A XML-based workflow event logging mechanism for workflow mining. *Lecture Notes in Computer Science, 3842,* 132-136.

Kim, K. (2006). An enterprise workflow Grid/P2P architecture for massively parallel and very large scale workflow systems. *Lecture Notes in Computer Science, 3842,* 472-476.

Kim, K. (1999). Actor-oriented Workflow Model. *Proceedings of the 2nd international symposium on Cooperative Database Systems for Advanced Applications*.

Kim, K. (2005). A process-driven e-business service integration system and its application to e-logistics services. *Lecture Notes in Computer Science, 3762*, 485-494.

Kim, K. (2005). A Process-Driven Inter-organizational Choreography Modeling System. *Lecture Notes in Computer Science, 3762*, 485-494.

Wainer, J., Kim, K., & Ellis, C. A. (2005). A workflow mining method through model rewriting. *Lecture Notes in Computer Science, 3706,* 184-191.

Kim, K., & Ahn, H. (2005). An EJB-based very large scale workflow system and its performance measurement. *Lecture Notes in Computer Science, 3739*, 526-537.

Kim, K., & Kim, H. (2005). A Peer-to-Peer workflow model for distributing large-scale workflow data onto Grid/P2P. *Journal of digital information management, 3*(2), 64-70.

Kim, K., Ahn, H., & Kim, C. (2005). SCO control net for the process-driven SCORM content aggregation model. *Lecture Notes in Computer Science, 3483,* 38-47.

Kim, K., Won, J., & Kim, C. (2005). A fragment-driven process modeling methodology. *Lecture Notes in Computer Science*, *3483,* 817-826.

Kim, K., Lee, J., & Kim, C. (2005). A real-time cooperative swim-lane business process modeler. *Lecture Notes in Computer Science, 3483,* 176-185.

Kim, K., Yoo, H., & Won, J. (2004). The e-Lollapalooza global workflow modeler: a registry-based e-Logistic. *Lecture Notes in Computer Science,* (pp. 419-430).

Kim, K. (2004). Cooperative fragment-driven workflow modeling methodology and system. *WfMC Workflow Handbook,* (pp. 189-207).

Kim, K., et al. (2003). Role-based model and architecture for workflow systems. *International Journal of Computer and Information Science, 4*(4).

Kim, K. (2003). Workflow dependency analysis and its implications on distributed workflow systems. *Proceeding of the AINA*, (pp. 677-682).

Kim, K. (2003). Workflow reduction for reachable-path rediscovery. *IEEE Proceeding of the ICDM Workshop.*

Kim, K., & Kim, I. (2002). The Admon-Time workflow client: why do we need the third type of workflow client designated for administration and monitoring services? *Lecture Notes in Computer Science, 2419,* 213-224.

Kim, K., & Ellis, C. A. (2001). Performance analytic models and analyses for workflow architectures. *Journal of Information Systems Frontiers, 3*(3), 339-355.

ENDNOTES

1 The work was supported by the Contents Convergence Software Research Center, 2007-81-0, funded by the GRRC Program of Gyeonggi Province, South Korea.

2 In terms of the terminological usage, workflow procedure can be interchangeably used with business process. We prefer workflow procedure to business process in this chapter.

3 Workflow Process Definition Language

4 XML-based Process Definition Language, Version 1.0 and 2.0 are available in the international standardization organization, workflow management coalition.

5 The terminology and concept of block was firstly used in the workflow modeling system of FlowMark Workflow Management System of IBM.

6 Workflow Process Definition Language

7 XML-based Process Definition Language

8 P2P stands for Peer-to-Peer grid computing environment.