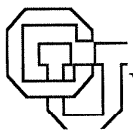
Actor Oriented Workflow: A Concept and Its Formalization

Kwang-Hoon Kim
Clarence A. Ellis

CU-CS-848-97



University of Colorado at Boulder
DEPARTMENT OF COMPUTER SCIENCE

Actor Oriented Workflow: A Concept and Its Formalization

Kwang-Hoon Kim Clarence A. Ellis

CU-CS-848-97 OCTOBER 1997

University of Colorado at Boulder

Technical Report CU-CS-848-97
Department of Computer Science
Campus Box 430
University of Colorado
Boulder, Colorado 80309-0430

Actor Oriented Workflow: A Concept and Its Formalization

Kwang-Hoon Kim Clarence A. Ellis

OCTOBER 1997

Abstract

This paper formally defines a multiple-level abstraction of workflow model in order to provide a theoretical basis for actor oriented distribution in designing and implementing a workflow management system. Our key contribution in this paper is to propose a workflow architecture in which all information needed by a user (called an actor) is downloaded to his/her workstation in advance. This is radically different from the typical client-server workflow architecture. The means of gathering and downloading exactly the right information (but no more), is elucidated by our formalism. This work is based upon the strong belief that the concept of actor orientation in workflow management might be effective and efficient considering recent trends in the technology as well as the organizational environment for workflow management. In other words, the personal computing facility has gradually become more powerful, and workflow procedures (or business processes) have gradually become more complex and larger. So, distribution in terms of workflow architectures for enactment control and information management must be a key ingredient in the solution for these new workflow management environments. The embodiment of a workflow architecture is closely related to its workflow model representation and to how the model is concretized in the architecture. Therefore, we are proposing a workflow model pursuing actor oriented distribution, which is re-constructed through the formally defined multiple-level abstraction of the information control net.

The formal abstraction consists of view-level, conceptual-level and physical-level workflow. The view-level workflow is represented by the extended information control net. For this, we extend the information control net(ICN), which is a typical workflow model for specifying workflow procedures in organizations, so as to represent actors and repositories associated with activities on a workflow procedure. It is called the extended information control net. The conceptual-level workflow is represented by an actor-oriented net consisting of actor-dependent net, control-dependent net and data-dependent net. These dependent nets are able to be generated through dependent analysis in workflow model, and their construction algorithms are specified. The physical-level representation of a workflow procedure is realized in a workflow management system through a set of actor transition conditions used to control the enactment transitions among actors being involved in the workflow procedure.

Finally, we try to describe the possible usability of the workflow model abstraction for solving the long-term and short-term workflow research issues, such as dynamic changes in workflow, transactional workflow, dynamically evolving workflow, large-scale workflow, etc., that have been proposed in the literature.

Key words: *Workflow Management Systems, Workflow Models, Actor Orientation, Extended Information Control Nets, Workflow Model Abstraction*

1. Introduction

In general, a workflow management system consists of two components: Modeling and Enactment. The Modeling component allows a modeler to define, analyze, and maintain all the information necessary to eventually execute workflow procedures. The workflow procedures are finally performed by the Enactment component of the workflow management system. The conceptual part, i.e. the logical foundation of the Modeling component, is the workflow model. It contains all objects and relationships available to describe a particular workflow. The expressiveness of a workflow management system is decided by the content of this model.

The workflow model has a lot of influences on the design and implementation of a workflow management system. That is, according to the workflow model, the corresponding workflow management system may have different features as well as different efficiency. Therefore, the workflow model should be able to incorporate the advanced technological and organizational trends so that the corresponding workflow management system enables to be effective and acclimatizes itself to a new technological and organizational environment. The recent trends in workflow management, which we have to take into account, are increasingly powerful networked personal computing facilities, and increasingly large and complex workflow applications.

So far, several types of workflow models have been introduced in the literature of workflow management, such as the communication-based model, activity-based model, perspective-based model and transactional model. These workflow models support the specification of workflow procedures to represent the structures of office procedures and to capture their information in an organization. So, the workflow model becomes a theoretical and practical basis for designing a workflow management system. Considering the advanced workflow requirements[15], such as dynamic changes, goal-based, and distributed and large scale, a workflow model should provide a way to effectively represent flexibility, distribution and scalability aspect of workflow procedures, corresponding to dynamically evolving workflow procedures, distributed enactment control and an increasing number of actors, respectively. Previous workflow models lack support for these aspects[7][10]. So, we expect that this actor oriented workflow should be a possible solution for those requirements.

We strongly believe that those workflow models can be more effective and efficient if the concept of actor orientation is added into them, because the main tasks of workflow management are to coordinate and support human activities in an organization. This paper specifically addresses the Information Control Net (ICN)[4][8]. ICN systematically and formally formulate a way to describe and incorporate the concept of actor orientation into the workflow models.

In the following section, we briefly describe typical existing workflow models and re-define the workflow components and their relationships by the entity-relationship diagram to understand the workflow model with priority given to actors. Next, we introduce a multiple-level abstraction of the ICN workflow model and specify it in a formal manner. Finally, the implication of the actor orientation in workflow is described in the end of the paper.

2. Workflow Models

To perform its task that helps to define, execute, coordinate, and monitor the flow of the work within organizations or work-groups, the workflow management system needs to be supplied with a computerized representation of the structure of office procedures and activities. This is a workflow model which is centered on actors as shown in FIGURE 1 using the entity-relationship model.

Amongst the components of the workflow model, ACTOR is the central element for performing workflow management. An ACTOR is a person, COMPUTING FACILITY, or GROUP of people that can fulfill ACTIVITIES which are associated with several different PROCEDURES. The basic concepts of the components in FIGURE 1 are well described in [4][14].

The currently available workflow management systems have been based upon the workflow models briefly described as the following:

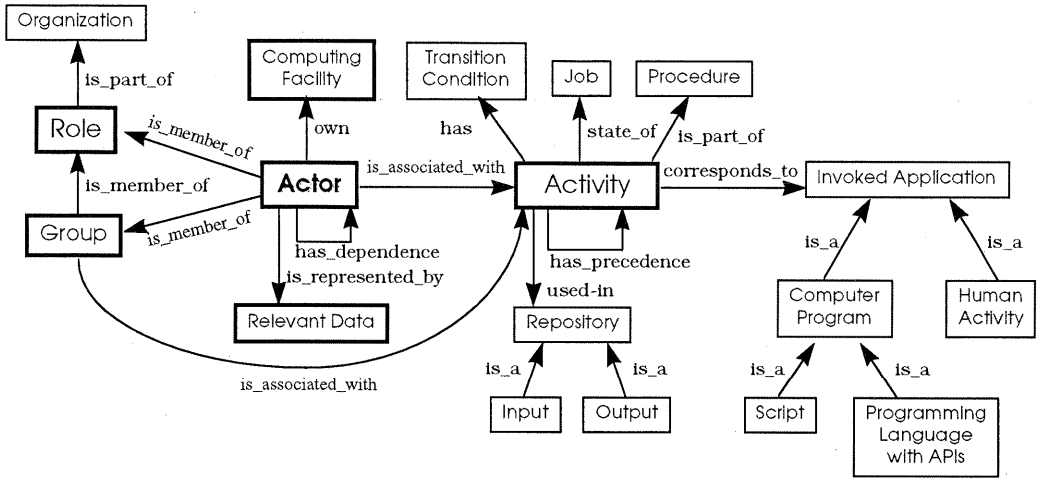


Figure 1. Actor-centered E-R Model of Workflow Components

Communication-based Model [12]: 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 [5][8][14]: 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. 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 [11]: 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 [9]: 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.

3. A Formal Abstraction for Actor Oriented Workflow

Between the workflow management system, dealing with transition conditions for controlling flows of

control and data within a procedure, and the ultimate user dealing with office procedures representing the flow of control and data, there are several levels of abstraction. A fairly reasonable viewpoint regarding levels of abstraction for the actor orientation in workflow model is shown in FIGURE 2: View Level, Conceptual Level, and Physical Level workflow.

The collection of office procedures, which are represented by the extended ICN(Information Control Net) model scheme - integrated the actors associated with each activity into the original ICN, is termed a *view level of workflow*. The view level workflow is handled by the modeling part of the workflow management system.

A set of transition conditions, constructed on each activity, for the control flow and data flow in a procedure is termed a *physical level workflow*. So, the transition conditions consist of actor-transition conditions, data-transition conditions, and control-transition conditions, corresponding to the actor flow, the data flow, and the control flow within a procedure, respectively. Each activity must be satisfied with both of its transition conditions in order to be enacted. The physical level workflow is used by the enactment part of the workflow management system.

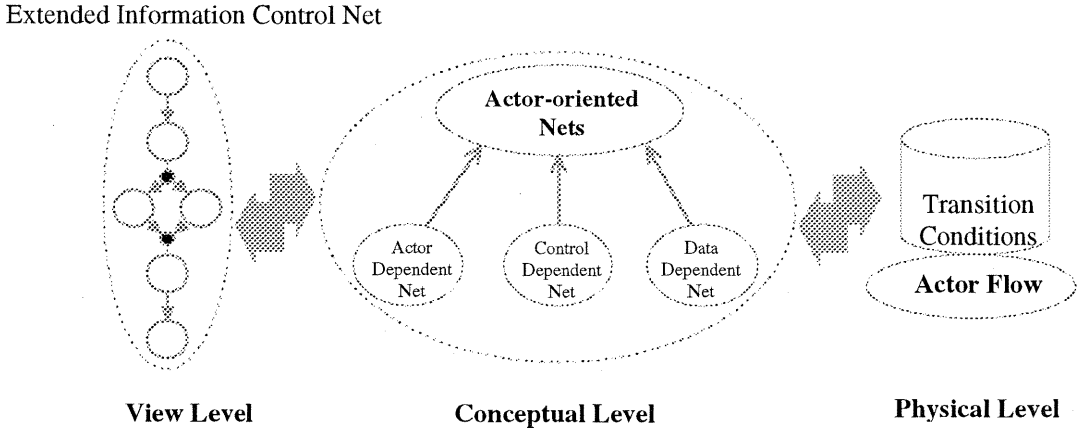


Figure 2. Levels of Abstraction for Actor Oriented Distribution

The *conceptual-level workflow* is an abstraction of the real actor orientation in workflow model. The conceptual level workflow is the intermediate phase to generate the physical level workflow from the view level workflow. That is, an actor dependent net, a control dependent net, and a data dependent net, that make up an actor-oriented net, are generated from the extended ICN. And from the actor-oriented net, the transition conditions for the data flow and the control flow are generated. The following sections explain about each of the abstraction levels, in detail.

3.1 The View Level of Workflow: Extended Information Control Nets

The original Information Control Net [6] was developed by researchers(C.A.Ellis, G. Nutt, *et el.*) at Xerox PARC to describe and analyze information flow within offices. It has been used within actual as well as hypothetical automated offices to yield a comprehensive description of activities, to test the underlying office description for certain flaws and inconsistencies, to quantify certain aspects of office information flow, and to suggest possible office restructuring permutations.

The ICN model defines an office as a set of related procedures. Each procedure consists of a set of activities connected by temporal orderings called procedure constrains. In order for an activity to be accomplished, it may need information from repositories, such as files, forms, and some data structures. An ICN captures these notations of procedures, activities, precedence, and repositories in graphical forms. In order to support the actor orientation in workflow model, we extend the basic ICN model so as to incorporate the notion of repositories and actors (including actor-groups) as suggested in [17]. Our

extended ICN definition fits the ICN schema defined in [6], and is thus a valid member of the ICN family of models.

3.1.1 Definition

An extended ICN is 5-tuple $\Gamma = (\delta, \gamma, \pi, I, O)$ over a set A of activities (including a set of group-activities), a set R of repositories, 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 δ_o: $A \rightarrow \wp(A)$ is a multi-valued mapping of an activity to its sets of (immediate) successors, and δ_i: $A \rightarrow \wp(A)$ is a multi-valued mapping of an activity to its sets of (immediate) predecessors. (For any given set S, $\wp(S)$ denotes the power set of S.)
- $$\gamma = \gamma_i \cup \gamma_o$$

  where γ_o: $R \rightarrow \wp(A)$ is a multi-valued mapping of an activity to its set of output repositories, and γ_i: $R \rightarrow \wp(A)$ is a multi-valued mapping of an activity to its set of input repositories;
- $$\pi = \pi_a \cup \pi_c$$

  where π_a: $C \rightarrow \wp(A)$ is a multi-valued mapping of an activity to its sets of associated actors, where the actors assigned into an activity have a relationship represented by the combination of AND($\wedge$) and OR($\vee$), and π_c: $A \rightarrow \wp(C)$ is a multi-valued mapping of an actor to its sets of associated activities;

FIGURE 3 is an example modeled by the extended information control net. This workflow procedure is simply taken out of an order processing workflow procedure in [5] with priority given to simplification. As shown in the figure, it is possible for multiple actors with or/and relationships to be assigned into activities. Based on this extended ICN model, we are going to construct control, data and actor dependent net as an example.

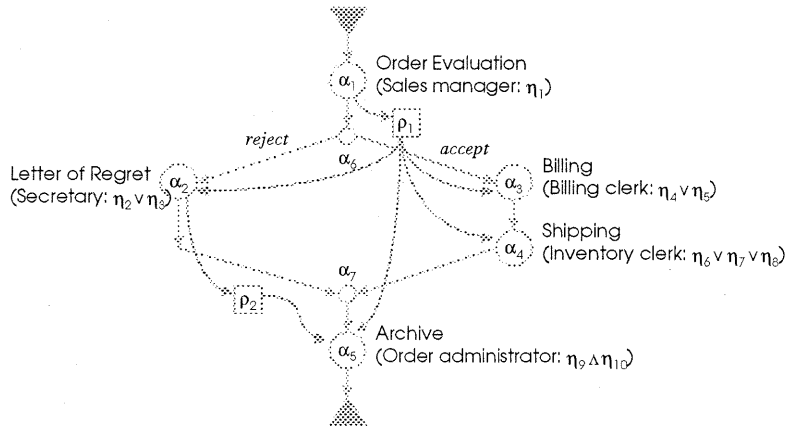


Figure 3. An Example of Extended ICN

3.1.2 Terminology

From the extended information control net, the following two terms, which are originally defined in [2], are able to be re-defined so as to be fitted into the workflow model. These terms are used in generating a control dependent net and a data dependent net from an extended information control net.

(1) A *walk*, W, in an extended ICN

In an extended ICN, $\Gamma = (\delta, \gamma, \pi, \kappa, \boldsymbol{I}, \boldsymbol{O})$, a *walk* $\boldsymbol{W}$ is a sequence of activities, $\alpha_1, \alpha_2, \ldots, \alpha_n$, such that $n \geq 0$ and $\alpha_{i+1} \in \delta_o(\alpha_i)$ for $i = 1, 2, \ldots, n-1$. The length of a *walk* $\boldsymbol{W} = \alpha_1\alpha_2 \ldots\alpha_n$, denoted $|\boldsymbol{W}|$, is the number n of activity occurrences in $\boldsymbol{W}$.

Note that a walk of length zero has no activity occurrences; such a walk is called *empty*. A nonempty walk whose first activity is u and whose last activity v is called a $u - v$ *walk*. If $W = w_1w_2 \ldots w_m$ and $X = x_1x_2 \ldots x_n$ are walks such that either W is empty, X is empty, or w_m is adjacent to x_1, then the concatenation of W and X, denoted WX, is the walk $w_1w_2 \ldots w_mx_1x_2 \ldots x_n$.

(2) Dominance in an extended ICN

An extended ICN, Γ, satisfies each of the following conditions: 1) Γ contains two distinguished activities: the initial activity α_I, which $\delta_i(\alpha_I)$ is $\varnothing$, and the final activity α_F, which $\delta_o(\alpha_F)$ is $\varnothing$; 2) Every activity of Γ occurs on some $\alpha_I - \alpha_F$ walk.

- Let Γ be an extended ICN. An activity $u \in \boldsymbol{A}$ *forward dominates* an activity $v \in \boldsymbol{A}$ iff every $v - v_F$ walk in Γ contains u; u *properly forward dominates* v iff $u \neq v$ and u forward dominates v.
- Let Γ be an extended ICN. An activity $u \in \boldsymbol{A}$ *strongly forward dominates* an activity $v \in \boldsymbol{A}$ iff u forward dominates v and there is an integer $k \geq 1$ such that every walk in Γ beginning with v and of length $\geq k$ contains u.
- Let Γ be an extended ICN. The *immediate forward dominator* of an activity $\alpha \in (\boldsymbol{A} - \{\alpha_F\})$, denoted $ifd(\alpha)$, is the activity that is the first proper forward dominator of α to occur on every $\alpha - \alpha_F$ walk in Γ.

3.2 The Conceptual Level of Workflow

3.2.1 Control Dependent Nets

A control dependent net is used to model the effect of conditional branch on the behavior of procedures. Through the control dependent net, it is possible to efficiently generate control-transition conditions on each activity. These control-transition conditions are used for the flow of control among activities as well as the dynamic changes on the flow by the enactment part of the workflow management system. It is constructed out of the set δ(control flow part) in an extended ICN model.

3.2.1.1 Definition

A control dependent net is formally defined as $\boldsymbol{\Omega} = (\varphi, \kappa^c, \boldsymbol{S}, \boldsymbol{E})$, over a set $\boldsymbol{A}$ of activities and a set $\boldsymbol{T}$ of control-transition conditions, where

- $\varphi = \varphi_i \cup \varphi_o$

  where φ_o: $\boldsymbol{A} \rightarrow \wp(\boldsymbol{A})$ is a multi-valued mapping of an activity to a set of activities which are control dependent on it, and

  φ_i: $\boldsymbol{A} \rightarrow \wp(\boldsymbol{A})$ is a multi-valued mapping of an activity to a set of activities on which it is control dependent;

- *control-transition conditions* : $\kappa^c = \kappa^c_i \cup \kappa^c_o$

  where, κ^c_i: a set of control-transition conditions ($\in \boldsymbol{T}$) between $\varphi_i(\alpha)$ and α; and

  κ^c_o: a set of control-transition conditions ($\in \boldsymbol{T}$) between α and $\varphi_o(\alpha)$), where $\alpha \in \boldsymbol{A}$;

- $\boldsymbol{S}$ is a finite set of initial control-transition conditions, assumed to be loaded with control information by some external process before execution of the ICN;
- $\boldsymbol{E}$ is a finite set of final control-transition conditions, perhaps containing control information used by some external process after execution of the ICN.

3.2.1.2 Construction of Control Dependent Net

An control dependent net is constructed from an extended information control net through the following algorithm. For the algorithm, we need the concepts of the strongly control dependent which is defined in [2]. (Note that we need to remember the terms, walk and dominance.)

Let Γ be an extended information control net, and let $u, v \in A$. Then activity u is *strongly control dependent* on activity v iff there exists a v–u walk vWu not containing the immediate forward dominator of v.

Note that the activities that are strongly control dependent on a OR-control construct are those in between the fork activity and the join activity of the OR-control construct. Additionally, the activities that are strongly control dependent on the branch condition of a loop-control construct are the control-transition conditions themselves and the activities in the inside of the loop. In the algorithm, the function, DECIDE( ), decides control-dependent conditions between two activities by taking sequential-control, AND-control and OR-control constructs into account.

```
INPUT : An extended ICN;
OUTPUT : A control dependent net;
BEGIN
 FOR all x ∈ A and y ∈ A in an extended ICN
     IF x is strongly control dependent on y        // OR or Loop Constructs
          ADD x TO φo(y);        ADD y TO φi(x);    // Get a dependent flow
     ELSEIF x is immediate forward dominator of y AND |xWy| is equal to 1
          ADD x TO φo(y);    ADD y TO φi(x);        // Get a dependent flow
     FI
     tempConTran ← DECIDE(Control-transition Conditions between x and y);
     κ^c_o(y) ← tempConTran;        // Get control-transition conditions between y and x
     κ^c_i(x) ← tempConTran;
 ROF
END.
```

The control dependent net construction algorithm for an extended information control net can be computed in $O(|A|^4)$. Since the algorithm for the strongly control dependent relations can be computed in $O(|A|^3)$[2], the algorithm for the immediate forward dominator relations can be computed in $O(|A|)$, so the inside of the for-loop in above algorithm can be computed in $O(|A|^3)$, and the for-loop itself can be done in $O(|A|)$, therefore, the overall computation time is $O(|A|^4)$.

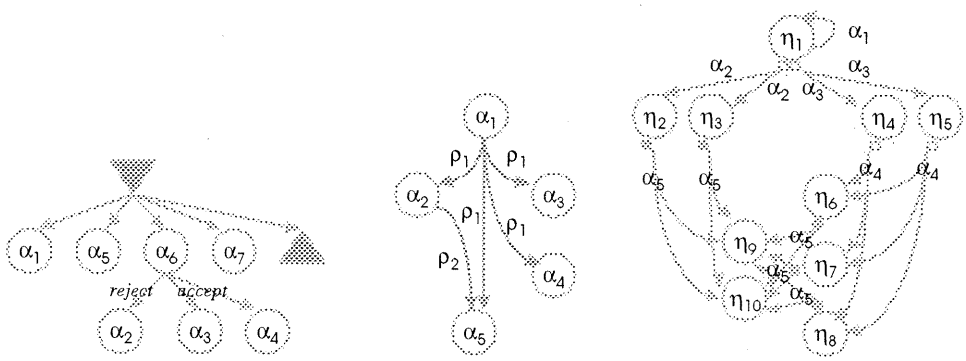


Figure 4. Control-Dependent, Data-Dependent, Actor-Dependent Net of the Example

The left-side directed graph in FIGURE 4 represents the control dependent net for the example extended ICN model. It shows the dependencies among the activities with control-transition conditions(*reject, accept*). Especially, the enactment of activity, α_2, depends on the activity, α_6, with respect to '*reject*' control-transition condition. Also, the activities, α_3 and α_4, relies on the activity, α_6, with respect to

'*accept*' control-transition condition.

3.2.2 Data Dependent Nets

A data dependent net is used to model the effect of data flow among activities on the behavior of procedures. Through the data dependent net, it is possible to efficiently generate data-transition conditions on each activity as well as data dependence information between activities. The data-transition conditions are used for activities to be enabled by the enactment part of the workflow management system. Even though the control-transition conditions on an activity are ready, it is not able to be enacted without getting the data-transition conditions ready. It is constructed out of the set γ(data flow part) in an extended ICN model. (Note that the notations and their meanings are same to that are used in the extended ICN's definition, if they are not re-defined.)

3.2.2.1 Definition of Data Definition–Use Net

A data def-use net is formally defined as $\Phi = (\delta, \upsilon, I, O)$, over a set A of activities and a set R of repositories, 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 $\delta_o: A \rightarrow \wp(A)$ is a multivalued mapping of an activity to its sets of (immediate) successors, and $\delta_i: A \rightarrow \wp(A)$ is a multivalued mapping of an activity to its sets of (immediate) predecessors;
- $$\upsilon = \upsilon_d \cup \upsilon_u$$

  where $\upsilon_d : A \rightarrow \wp(R)$ is a single-valued mapping function, and sets of defined(output) repositories, $\rho \in R$, on each activity, $\alpha \in A$, and $\upsilon_u : A \rightarrow \wp(R)$ is a single-valued mapping function, and sets of used(input) repositories, $\rho \in R$, on each activity, $\alpha \in A$.

3.2.2.2 Construction of Data definition-use Net

A data def-use net is constructed from an extended information control net through the following simple algorithm;

```
INPUT : An extended ICN;
OUTPUT : A data def-use net;
BEGIN
    GET a control flow subnet of the extended ICN, δ = δi ∪ δo;
    FOR all x ∈ A in an extended ICN
        ADD γi(x) TO υu(x) ;        // Get a set of use repositories on activity x
        ADD γo(x) TO υd(x) ;        // Get a set of def repositories on activity x
    ROF
END.
```

The data def-use net construction algorithm for an extended information control net can be computed in $O(|A|)$.

3.2.2.3 Definition of Data Dependent Net for a specific repository

A data dependent net is constructed by being based on a repository, $\rho \in R$. Therefore, there are $|R|$ data dependent nets in an extended ICN. A data dependent net for $D \in R$ is formally defined as $\Re = (\chi, \kappa^d, D)$, over a set A of activities and a set R of repositories, where

- D is a repository;

- $\chi = \chi_i \cup \chi_o$

  where χ_o: $A \rightarrow \wp(A)$ is a single-valued mapping of an activity to a set of activities depending on it with respect to D, and χ_i: $A \rightarrow \wp(A)$ is a single-valued mapping of an activity to a set of activities on which it depends with respect to D.

- *data-transition conditions* : $\kappa^d = \kappa^d_i \cup \kappa^d_o$

  where, κ^d_i: a set of data-transition conditions ($\in R$) between $\varphi_i(\alpha)$ and α; and

  κ^d_o: a set of data-transition conditions ($\in R$) between α and $\varphi_o(\alpha)$), where $\alpha \in A$;

3.2.2.4 Construction of Data Dependent Net

Before making an algorithm generating data dependent nets from a data def-use net, the following terms which are defined in [2] are needed: Let $\Phi = (\delta, \upsilon, I, O)$ be a data def-use net(graph), and let $u, v \in A(\Phi)$. Activity u is *directly data flow dependent* on activity v iff there is a walk $v\mathbf{W}u$ in Φ such that $(\upsilon_d(v) \cap \upsilon_u(u)) - \upsilon_d(\mathbf{W}) \neq \varnothing$; u is *data flow dependent* on v iff there is a sequence $\alpha_1, \alpha_2, \ldots, \alpha_n$ of activities, $n \geq 2$, such that $v = \alpha_1$, $u = \alpha_n$ and α_i is *directly data flow dependent* on α_{i+1} for $i = 1, 2, \ldots, n - 1$.

Note that if u is data flow dependent on v then there is a walk $\alpha_1\mathbf{W}_1\alpha_2\mathbf{W}_2 \ldots \alpha_{n-1}\mathbf{W}_{n-1}\alpha_n$, $n \geq 2$, such that $v = \alpha_1$, $u = \alpha_n$ and $(\upsilon_d(\alpha_i) \cap \upsilon_u(\alpha_{i+1})) - \upsilon_d(\mathbf{W}_i) \neq \varnothing$ for $i = 1, 2, \ldots, n - 1$. Such a walk is said to *demonstrate* the data flow dependent of u upon v.

A data dependent net for a repository is able to be constructed from a data def-use net by the following simple algorithm;

```
INPUT : A data def-use net, A set A of activities, A repository η ∈ R;
OUTPUT : A data dependent net for η;
BEGIN
    FOR all x∈ A and y ∈ A in a data def-use net
        IF x is data flow dependent on y with respect to a repository η
                ADD  x  TO  χo(y) ;
                ADD  y  TO  χi(x) ;
                ADD  η  TO  κ^d_o(y) ;   // get data-transition conditions
                ADD  η  TO  κ^d_i(x) ;
        FI
    ROF
END.
```

The data dependent net construction algorithm for all repositories in an extended information control net can be computed in $O(|R| \times |A|^2)$. Since the algorithm for the data flow dependent relations can be computed in $O(|A|)$ and the for-loop itself can be done in $O(|A|)$, the computation time for a repository is $O(|A|^2)$. Because there are $|R|$ repositories in an extended ICN, the overall computation time is $O(|R| \times |A|^2)$.

The middle directed graph in FIGURE 4 is the data dependent net for the example extended ICN. The graph presents data dependencies among the activities with respect to the repositories, ρ_1 and ρ_2.

3.2.3 Actor Dependent Nets

An actor dependent net represents the order of activity enactment among actors within a procedure. It is constructed out of the sets δ(control flow part) and π(actor assignment part) in an extended ICN model. The actor dependent net represents three kinds of dependencies among actors: sequential dependency, OR-dependency and AND-dependency. Especially, the AND-dependency represents an actor-group assigned to a group activity directly related with groupware supports in workflow. Almost all current available

workflow models do not support a group of actors, who are simultaneously working together, on a single activity.

3.2.3.1 Definition

An actor dependent net is formally defined as $\Lambda = (\sigma, \psi, \kappa^a, I)$, over a set C of actors, a set X = {*singular*-dependency, OR-dependency, AND-dependency} of dependencies and a set A of activities, where

- $\sigma = \sigma_i \cup \sigma_o$

  where σ_o: $C \rightarrow \wp(C)$ is a multi-valued mapping of an actor to its set of (immediate) successors including itself, and σ_i: $C \rightarrow \wp(C)$ is a multi-valued mapping of an actor to its set of (immediate) predecessors.

- $\psi = \psi_i \cup \psi_o$

  where ψ_i: a set of pairs, $(\alpha \in A, \chi \in X)$, on each arc, $(\sigma_i(\eta), \eta)$; and ψ_o: a set of pairs, $(\alpha \in A, \chi \in X)$, on each arc, $(\eta, \sigma_o(\eta))$, where $\eta \in C$;

- *actor-transition conditions* : $\kappa^a = \kappa^a_i \cup \kappa^a_o$

  where, κ^a_i: a set of actor-transition conditions, a set of pairs, $(\alpha \in A, \chi \in X)$, between $\sigma_i(\eta)$ and η; and

  κ^a_o: a set of actor-transition conditions, a set of pairs, $(\alpha \in A, \chi \in X)$, between η and $\sigma_o(\eta)$), where $\eta \in C$;

- I is a finite set of coordinators or coordinator-groups, assumed that it has the responsibility to initiate or instantiate a procedure at execution of the ICN.

3.2.3.2 Construction of Actor Dependent Net

An actor dependent net is constructed from an extended information control net through the following simple algorithm;

```
INPUT : An extended ICN;
OUTPUT : An actor dependent net;
BEGIN
    FOR all x ∈ A in an extended ICN
        ADD all member of π_a(x) TO σ_i(each member of π_a(δ_o(x)));
        ADD all member of π_a(δ_o(x)) TO σ_o(each member of π_a(x));
        // Get an arc labeled by (activity, relationship) between two actors.
        ADD (x, relationship of π_a(x)) TO ψ_i(all members of π_a(δ_o(x)));
        ADD (x, relationship of π_a(δ_o(x)) TO ψ_o(all members of π_a(x));
    ROF;
    // Get actor-transition conditions
    FOR all y ∈ C in an extended ICN
        ASSIGN ψ_i(y) TO κ^a_i(y);
        ASSIGN ψ_o(y) TO κ^a_o(y);
    ROF;
END.
```

The actor dependent net construction algorithm for an extended information control net can be computed in $O(|A|)$.

The right-side directed graph in FIGURE 4 represents the actor dependent net for the example extended ICN. In the graph, the dependencies are represented by types of arrow; a directed arc without dot, a

directed arc with open-dot and a directed arc with filled-dot represent *singular*-dependency, OR-dependency and AND-dependency among actors, respectively.

3.3 The Physical Level of Workflow

Finally, an extended information control net is physically represented and transformed into a set of transition conditions consisting of actor-transition conditions, control-transition conditions, and data-transition conditions. FIGURE 5 illustrates how the transition conditions are represented in the physical level workflow, and related with each other.

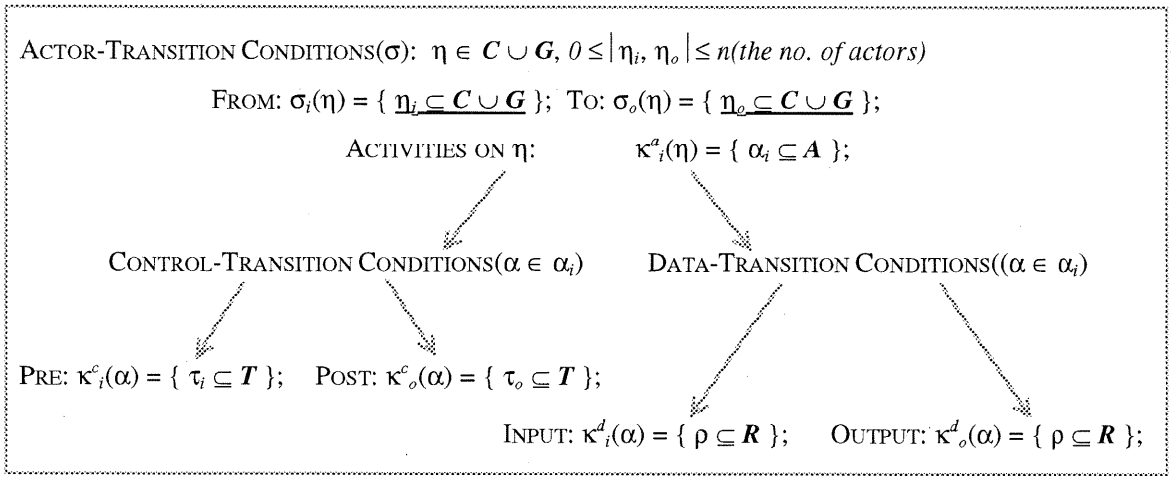


Figure 5. Transition Conditions on Actor Oriented Workflow Model

In the enactment part of the workflow management system, the activity enactment can be managed through the data, control, and actor flow information. That is, an activity on an office procedure can be executed if the following information is ready:

- Who is assigned to do the activity? : *actor-transition conditions*
- Does the activity have the control? : *control-transition conditions*
- Is the input-data ready? (Optional) : *data-transition conditions*

This information can be gained easily from the actor oriented net of a workflow model as shown in FIGURE 5. The actor-transition conditions come from an actor-dependent net, the control-transition conditions come from a control-dependent net, and the data-transition conditions are generated by a data def-use/dependent net. In other words, actors are able to get those activities that should be done by themselves, those control-transition conditions that must be ready for each activity to be enacted, and those input data that should be provisioned for each activity to be enacted.

In general, the assignment of performers into an activity is based on roles and fulfilled at the modeling time. A role is a named designator for an actor, or a grouping of actors which conveniently acts as the basis for access control and execution control. A real actor out of actors who are associated with the role is selected right after the activity is enacted by a scheduler or an enactment controller in the enactment time. We would refer to the actor assignment scheme like this as '*the late actor binding*' scheme. As the opposite scheme, we propose '*the early actor binding*' scheme.

In the early actor binding scheme, the actor assignments into all activities of a workcase or an instance of a workflow procedure are fulfilled right after a workflow procedure is created, or the workcase or the instance is created. The former is called the *static actor assignment*, the latter is called the *dynamic actor assignment*. Then, it should be expected to do the efficient assignment of actors. These actor assignment schemes are able to be implemented by constructing a *static* actor dependent net and a *dynamic* actor dependent net, respectively.

A static actor dependent net is constructed only once during the whole life of a workflow procedure. So, an actor assigned to specific activities of a workflow procedure has to perform all workcases or instances corresponding to the activities. However, a dynamic actor dependent net is able to be constructed whenever a workcase or instance is created. Therefore, it is possible to accomplish a dynamic actor assignment based on a workcase or instance of a workflow procedure. As a result, the actor oriented workflow model is able to support the late actor binding scheme as well as the early actor binding scheme.

4. Conclusions

This paper has introduced the concept of actor-oriented workflow model, and developed a formalism to transform an ICN model into an actor-oriented model. The important idea of the actor oriented workflow model is that an actor is able to do his/her activities on his/her own computing facility without being controlled by the system if the three transition conditions for the activities are ready. This means that, in the actor-oriented workflow model, it is possible for every actor to manage its own activities, even belonging to several different office procedures, on its own territory or computing facility. Therefore, the actor-oriented distributed enactment control can be realized in designing and implementing a workflow management system. The followings are some useful examples for the actor oriented workflow model to be applied:

- Actor oriented distribution in enacting a workflow. For an example, in the persistent message-based workflow architecture[17], which is a typical ongoing distributed workflow system, this model is effectively used to configure a distributed network for a workflow procedure.
- Information coming from the formal abstraction enables a workflow management system to support dynamic changes in a workflow procedure; That is, the information is able to be used for the system to check over the influence of the changes before it allows a user to do them.

As future research issues, there might be several extensions; we should investigate how the concept of actor orientation is incorporated into the other families of workflow models, especially a family of the system-oriented workflow model like the transactional workflow model. In conclusion, we believe that the actor-oriented workflow model should be taken into consideration in the design and implementation of future workflow management systems.

5. References

[1] "The Business Imperative for Workflow & Business Process Reengineering", A Special Advertising Section, Fortune, Feb. 1996

[2] Andy Podgurski and Lori A. Clarke, "A Formal Model of Program Dependencies and Its Implications for Software Testing, Debugging, and Maintenance", IEEE Trans. on SE, Vol. 16, No. 9, Sep. 1990

[3] Clarence A. Ellis, J. Gibbs, and G.L. Rein, "Groupware: Some issues and Experiences", Communication of the ACM, Vol. 34, No. 1, Jan. 1991

[4] Clarence A. Ellis, Gary J. Nutt, "Office Information Systems and Computer Science", ACM Computing Surveys, Vol. 12, No. 1, March 1980

[5] Clarence A. Ellis, Gary J. Nutt, "The Modeling and Analysis of Coordination Systems", University of Colorado/Dept. of Computer Science Technical Report, CU-CS-639-93, Jan. 1993

[6] Clarence A. Ellis, "Formal and Informal Models of Office Activity", Proceedings of the 1983 Would Computer Congress, Paris, France, April 1983

[7] Clarence A. Ellis, Keddara K and Rozenberg G., "Dynamic Change within Workflow Systems", Proceedings of the ACM SIGOIS Conference on Organizational Computing Systems, Milpitas, CA., 1995

[8] C.A. Ellis and J. Wainer, "Goal-based Models of Collaboration", Journal of Collaborative Computing, Vol. 1, No. 1, March 1994

[9] Diimitrios Georgakopoulos, Mark Hornick, "An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure", Distributed and Parallel Databases, 3, pp. 115-153, 1995

[10] Gustavo Alonso and Hans-Joerg Schek, "Research Issues in Large Workflow Management Systems", Proceedings of NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-Art and Future Directions, May 1996

[11] Stefan Jablonski and Christoph Bussler, "Workflow Management: Modeling Concepts, Architectures and Implementation", International Thomson Computer Press, 1996

[12] James H. Bair, "Contrasting Workflow Models: Getting to the Roots of Three Vendors", Proceedings of International CSCW Conference, 1990

[13] Kwang-Hoon Kim, "Groupware: Introduction and State-of-the-Art Research and development", The Journal of Sciences, Kyong-Gi University, 1995

[14] Kwang-Hoon Kim, "Practical Experience on Workflow: Hiring Process Automation by FlowMark", IBM Internship Report, IBM/ISSC Boulder Colorado, 1996

[15] Kwang-Hoon Kim and Su-Ki Paik, "Practical Experiences and Requirements on Workflow", Lecture Notes Asian '96 Post-Conference Workshop: Coordination Technology for Collaborative Applications, The 2nd Asian Computer Science Conference, Singapore, 1996

[16] Clarence A. Ellis, Gary J. Nutt, "Modeling and Enactment of Workflow Systems", University of Colorado/Department of Computer Science, Technical Reports, March 1993

[17] A.D. Agrawal, et al, "Exotica/FMQM: A Persistent Message-based Architecture for Distributed Workflow Management", Proceedings of IFIP Working Conference on Information Systems for Decentralized Organizations, Aug. 1995