KWM : Knowledge-based Workflow Model for Agile Organization

Corresponds to :

Prof. Sung Joo Park
Graduate School of Management,
Korea Advanced Institute of Science and Technology,
373-1 Kusong-dong, Yusong-gu,
Taejon, 305-701, S. Korea
Tel. +82 42 869 2913
Fax. +82 42 869 2910
Internet: sjpark@cais.kaist.ac.kr

KWM : Knowledge-based Workflow Model for Agile Organization

Ha Bin Lee*, Jong Woo Kim**, Sung Joo Park*

*Graduate School of Management, Korea Advanced Institute of Science and Technology
** Department of Statistics, Chungnam National University

Abstract

The workflow management system (WFMS) in an agile organization should be highly adaptable to frequent organizational changes. To increase the adaptability of contemporary WFMSs, a mechanism for managing changes on the organizational structure and business rules need to be enhanced. In this paper, a knowledge-based approach for workflow modeling is proposed, in which a workflow is defined as a set of business rules. Knowledge on organizational structure and special workflow, such as role/actor mapping and complex routing rules, can be explicitly modeled in KWM (Knowledge-based Workflow Model). Using knowledge representation scheme and dependency management facility, change propagation mechanism is provided to adapt to frequent changes on organizational structure, business rules and procedures.

Key words: Workflow management, agile organization, adaptive workflow, business rule, knowledge-based system, change management.

1. Introduction

Increasing agility of an organization is considered as a critical success factor in a competitive environment of continually and unpredictably changing customer opportunities (Goldman et al., 1995). Agile organizations are apt to frequently change their business processes to satisfy fluctuating customers needs. The workflow management system (WFMS) as a technology that automates business processes should be highly adaptive to changes on business processes in agile organizations. In a WFMS, business processes are represented using workflow model which has three main constructs; routes, rules, and roles (Marshak, 1994). Routing construct represents task sequences and a role represents one who is responsible for a task. Based on organizational model, a role can be defined with actor's department, position, and skills, etc. Rule is used to define routing and role constructs. It enables to define conditional or exceptional routings and conditional assignment of tasks to actors through role constructs. An adaptive WFMS should be flexible enough to handle the changes on these three constructs.

Some WFMSs are flexible (Reichert and Dadam, 1998; Casati et al., 1998; Dellen et al., 1997) in the sense that they provide adaptability for the changes on routing constructs such as adding or deleting tasks, or changing task sequences. These systems, however, do not provide capability to handle changes on the organizational structure and business rules. The role definition can be affected by the changes on the organizational structure such as the merger and abolition of departments, change on the position hierarchy, and creation of temporal task force, etc. In an organization, there may exist heterogeneous departments and actor types, different routing conditions according to the types of actors, and flexible role instances that are responsible for a task. The rules change frequently due to BPR (Business Process Reengineering), empowerment, or restructuring. The existence of exceptional rules that may be applied for special workflow instances aggravates the complexity of rule management. The business rules can be

directly affected by the changes on the routing and role constructs, of which the effects can also be cascaded, i.e., change on a business rule can affect other related business rules. Thus providing a change propagation facility for the changes on the three constructs is an inevitable component of adaptive WFMSs.

For an ideal adaptive WFMS for agile organizations, workflow models need to be enhanced in the following aspects:

- Expressiveness : It should provide constructs to represent conditional mapping relationships between roles and actors based on organizational model as well as complex business rules including exceptional rules.

- Model verification : It should allow analysis that assures the correctness of workflow specification including checking the occurrence of inconsistent, redundant, and incomplete business rules as well as non-terminality of processes.

- Change management : It should allow easy development of propagation mechanism against changes on the organizational structure and business rules as well as organizational procedures to assure the correctness of workflow model.

In this paper, a knowledge-based approach for workflow modeling and enactment is proposed. KWM (Knowledge-based Workflow Model) is designed and implemented to enhance the three aspects. First, the expressive power of business rules of KWM are improved using knowledge-based approach to represent complex and heterogeneous business rules. Secondly, properties that assure correctness of KWM are proposed which can be analyzed using a rule verification technique. Thirdly, management of organizational changes in KWM can be easier due to change propagation mechanism. Dependencies between modeling constructs are explicitly represented in KWM, and organizational changes that affect routes, rules, and roles in a workflow are propagated to corresponding constructs using the dependencies

to assure the correctness of KWM.

This paper is composed as follows: Section 2 briefly reviews related research. The detail of KWM is described in section 3. In section 4, properties for the correctness of KWM are introduced, and an algorithm for checking the properties is described. A change propagation mechanism for KWM is presented in section 5. In section 6, KWM is applied to an example. Finally, section 7 concludes the paper.

2. Review of Related Research

Many works for workflow modeling are based on the input-process-output (IPO) approach (Gruhn, 1995; Ellis and Nutt, 1993; Wolf and Reimer, 1996; van der Aalst, 1998). It provides task-oriented view on workflows, that is, a workflow is considered as a set of interrelated tasks which process inputs and produce outputs. This approach is good to model structured workflows such as business trip approval process and purchasing process. On the other hand, language / action approach is also used for workflow modeling (Winograd, 1987; Flores et al., 1988; Michelis and Grasso, 1994; Kaplan et al., 1992). It is based on the conversations between workflow participants, and has merits for modeling unstructured workflow such as project planning. Some research employ object-oriented approach for workflow modeling and enactment (Bose, 1996; Chang and Scott, 1996; Jennings et al., 1996). Bose (1996) presented five classes of objects as a key construct: roles, organization structures, procedures, transitions, and documents. In his model, workflows are executed through message passing between participating objects of the workflows. Both of Chang and Scott (1996) and Jennings et al. (1996) suggested agent based approach for workflow management. In their architecture, autonomous and problem solving agents

interact via their own protocol to achieve workflow management goals.

In this paper, workflow is defined as a set of business rules. Business rules that control scheduling tasks and role/actor mapping are explicitly represented using knowledge representation scheme and a workflow is executed by firing the rules. The rule-based approach for workflow modeling has advantages in expressive power, verification, and change propagation. Some researches use the rule-based approach for workflow modeling or enactment. Davulcu et al. (1998) use transaction logic (Bonner and Kiffer, 1994) for workflow modeling and analysis. The main focus of the researche is representing and analyzing workflows and it does not address rule change propagation mechanism and implementation details. Kappel et al. (1995) and Casati el al. (1996) use event-condition-action (ECA) rules provided by active database management systems for workflow modeling and enactment. However, they do not address rule management issues such as rule verification and rule change management.

Verification issues in conceptual workflow specifications using Petri-net theory are addressed in Hofstede et al. (1998), Adam et al (1998), and Van der Aalst (1998). It is possible to check termination of workflow and occurrence of dangling tasks using Petri-net but it is difficult to check the correctness of routing condition or mapping rule between role and actor. The rule-based approach for workflow modeling given in this paper enables checking the correctness of the specification of routing conditions, the redundancy of rules, as well as the termination of workflow. Verification of a set of rules has been addressed in artificial intelligence (AI) field. Preece et al. (1992) summarize four main properties, i.e., redundancy, ambivalence, circularity, and deficiency, that should be checked for rule-base verification. They also compare some rule-base verification techniques that are used at expert system shells. Baralis and Widom (1994) suggest a propagation algorithm for verification of rule properties, i.e, termination and confluence, in expert database systems. In this paper, the soundness properties for workflow verification are defined based on the properties.

The issue of flexible workflow management has been addressed in Casati et al. (1998), Reichert and

Dadam (1998), Dellen et al. (1997), and Bogia and Kaplan (1995). Casati et al. (1998) suggested a set of primitives that allow modifications of workflow schema, and introduced a taxonomy of policies to manage the evolution of running instances when the corresponding workflow schema is modified. Reichert and Dadam (1998) defined a complete and minimal set of change operations ($ADEPT_{flex}$) that support users in modifying the structure of a running workflow while maintaining its structural correctness and consistency. Dellen et al. (1997) suggested CoMo-Kit system in which it is possible to refine and extend the software process model during process execution using dependency management and change notification mechanism. In these researches, managing the changes such as adding or deleting tasks and changing predefined task sequences are the main concern without considering mechanisms to handle changes on organizational structure and business rules.

3. KWM : Knowledge-based Workflow Model

The basic principles of designing Knowledge-based Workflow Model (KWM) are the flexibility of the model, the expressiveness for complex business rules, and the formality for enabling the analysis of workflow. In KWM, a workflow is defined as a set of business rules for scheduling of tasks, mapping role and actors and routing work items. Business rules restrict and guide a workflow execution according to the state of an organization. The state of organization is represented as a set of attribute values of the organizational objects. For effective modeling of business rules in workflow, two heterogeneous knowledge, i.e., declarative knowledge representing state of an organization and procedural knowledge representing state-based behavior, are represented using frames.

3.1 Formal Definition of KWM

Definition 1(workflow model) A KWM defines a workflow with a 3-tuple, $W = (E, Rl, Ru)$, where E is a set of entity frames and Rl is a set of relationship frames and Ru is a set of rule frames. A frame f in E, Rl, or Ru is defined as a product of slot and value pairs, that is,

$$f = (s_1, v_1) \times (s_2, v_2) \times \ldots\ldots \times (s_{n-1}, v_{n-1}) \times (s_n, v_n).$$

If two frames are the instances of the same class, these two frames have the same slots. Also, a slot of a frame can be a relationship of the frame, and the value of the slot can be another frame with which the frame has a relationship.

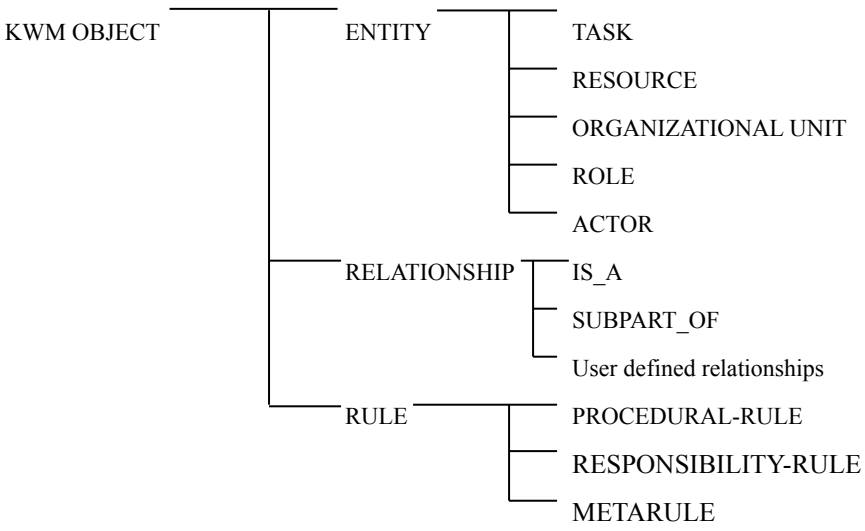

Figure 1. Hierarchy of frames in KWM.

Figure 1 is showing frame hierarchy in KWM. The basic specification syntax of a KWM frame is as follows;

<frame> ::= (<frame-identifier>, {<slot>})
<slot> ::= (<slot-spec>, <slot-value>)
<slot-spec> ::= (<attribute-name, <domain-type>) | 'CONDITION'
<slot-value> ::= <attribute-value> | {<condition-predicate>}

Definition 2 (Entity Frame) An entity frame in E belongs to one category among five kinds of objects; tasks, resources, organizational units, roles, and actors. That is, $E = T \cup Re \cup U \cup Ro \cup A$ where T is a set of tasks, Re is a set of resources, U is a set of organizational units, Ro is a set of roles, and A is a set of actors.

Definition 3 (Relationship Frame) A relationship frame is 3-tuple, $Rl = (s_o, s_i, P)$, where s_o is a source slot that contains source entity for the relationship, s_i is a sink slot that contains sink entity for the relationship, and P is a set of property slots of the relationship.

The entity frame is an abstraction of all entities in an organization and the relationship frame is an abstraction of the structural and behavioral relatedness between two entity frames. The entity and

```
FRAME Rule
  IS_A : KWM_object;
  PROCESS; <process-name>;
  DESCRITION: <string>;
  CONDITION : {<condition-predicate>};
END_FRAME

FRMAE Responsibility-Rule
  IS_A : Rule;
  ROLE: <role-entity>;
  ACTOR: <entity> '.' <slot>;
END-FRAME
```

```
FRAME Procedural_Rule
IS_A : Rule;
    PRE_TASK: <task-entity>;
    PRE_TASK_STATE: <task-state>;
    NEXT_TASK: <task-entity>;
END_FRAME

FRAME Metarule
    IS_A: Rule;
    SOURCE_RULE: {<rule-frame>,};
TARGET_RULE: {<rule-frame>,};
END-FRAME
```

Figure 2. The specification structure of rule frames

relationship frames contain information which is necessary to control workflow, that is, they are used to represent organizational model and resources. In the task set T, particularly, there are two artificial tasks called *Initiate* denoting the start of a workflow and *Terminate* denoting the end of a workflow.

Definition 4 (Rule Frame) A rule frame in *Ru* belongs to one category among three kinds of rules; procedural rules, responsibility rules, and metarules. That is, $Ru = Rp \cup Rr \cup Rm$ where *Rp* is a set of procedural rule, *Rr* is a set of responsibility rule, and *Rm* is a set of metarule. The set *Rp* is a set of rules that conditionally connect tasks with the tasks followed. The set *Rr* is a set of rules that conditionally relate roles with actors. The set *Rm* is a set of rules that conditionally relate two or more procedural rules or responsibility rules.

The rule frames contain rules that control execution of workflow based on the states of entity and relationship frames. Each rule frame contains multiple slots to represent attribute values for rule management purpose as well as condition and action parts of a rule. Figure 2 shows the specification structure of rule frames. Every rule frame is a sub class of the Rule frame with three slots; PROCESS, DESCRIPTION, and CONDITION. The PROCESS slot represents the process to which the rule is applied, and the DESCRIPTION slot represents verbal meaning of the rule. In the CONDITION slot, one or more condition predicates can be specified, and multiple condition predicates are connected with conjunctive relationship, that is, all the conditions should be satisfied to fire a rule.

The Procedural-Rule frames represent procedural view of a workflow. They define conditional sequences between tasks and also establish a communication network among actors in charge of tasks. In Figure 2, a Procedural-Rule frame illustrates that if the state of the task in the PRE_TASK slot is the value specified in the PRE_TASK_STATE slot, and all the conditions in the CONDITION slot are satisfied, then the task that is specified in the NEXT_TASK slot is followed.

The effective role modeling protects workflow model from the frequent organizational changes including the changes on the department hierarchy, employment or retirement of employees, and changes on the job position in an organization. The role concept is implemented with the Responsibility-Rule frames in KWM. The Responsibility-Rule frame guides workflow engine to find actors who are in charge of a role. In a Responsibility-Rule frame, the ACTOR slot contains a frame and a slot from which the

actor's identifiers can be extracted. The CONDITION slot contains constraints that instances of the frame specified in the ACTOR slot should satisfy.

Lastly, metarule frames are needed to handle exceptional rules. Exceptional rules are defined as the rules that are applied to special workflow instances. It is often prescribed how to handle special workflow instances in an agile organization. Exceptional rules are defined to handle special instances such as a business process for a special task force, temporary appointment to reduce overload of a special position, and emergency measure to process special customer needs, etc. Conceptually, a special workflow instance can be handled by substituting rules for the workflow instance. The specification structure of a metarule in Figure 2 represents that the set of rules specified in the SOURCE_RULE slot is substituted by the set of rules specified in the TARGET_RULE slot if the conditions specified in the CONDITION slot are satisfied for a workflow instance. For instance, frames mr1, mr2, and mr3 in figure 3 represent metarules

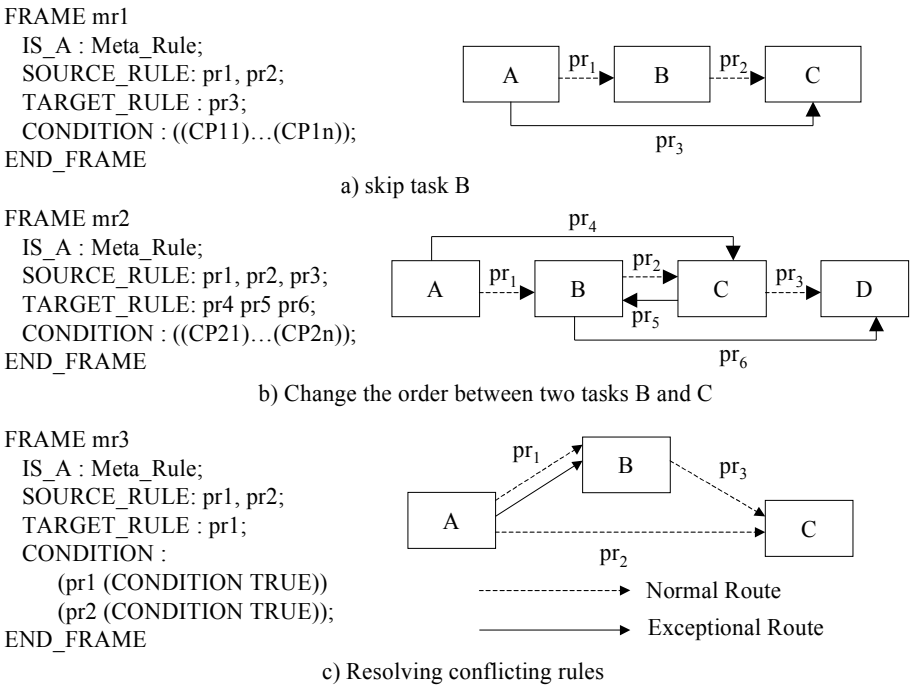

Figure 3. Metarule frames for procedural-rule frames

that handle exceptions for procedural rules. The metarule frame mr1 handles an exception that skips a task for a special workflow instance. The metarule frame mr2 is defined to change the order between two tasks. Lastly, the metarule frame mr3 is to resolve conflicts. It fires only procedural-rule frame pr1 when conditions of two procedural-rule frames (pr1 and pr2) are satisfied concurrently.

3.2 Routing Constructs

One of the main issues for workflow management is the routing of tasks to be executed. The workflow management coalition (WfMC) identified four routing constructs (WfMC, 1996). In KWM, the four routing constructs are represented using procedural-rule frames. The procedural-rule frames are equivalent to well-formed formulas (wffs) of the first order predicate calculus for abstract representation. Every rule in a procedural-rule set *Rp* can be represented as the following wff;

$$TS(x, C) \wedge CP_1 \wedge \ldots \wedge CP_n \Rightarrow TS(y,I)$$

Table 1. Representation of routing constructs using procedural-rule frame

Routing Constructs	Example
Sequential Routing	$\mathbf{Rp_1}$ = { TS(X, C) ⇒ TS(Y, I)}
Parallel Routing	$\mathbf{Rp_2}$ = { TS(W, C) ⇒ TS(X, I), TS(W, C) ⇒ TS(Y, I), TS(X, C) ∧ TS(Y, C) ⇒ TS(Z, I) }
Conditional Routing	$\mathbf{Rp_3}$ = { TS(W, C) ∧ COND1 ⇒ TS(X, I), TS(W, C) ∧ ¬COND1 ⇒ TS(Y, I), TS(X, C) ⇒ TS(Z, I), TS(Y, C) ⇒ TS(Z, I) }
Iterative Routing	$\mathbf{Rp_4}$ = { TS(X, C) ∧ COND2 ⇒ TS(Y, I), TS(X, C) ∧ ¬COND2 ⇒ TS(X, I) }

The TS predicate, meaning "task state", contains two terms representing a task and a state of the task, respectively. The two terms are taken from the PRE-TASK slot and the PRE-TASK-STATE slot of a procedural-rule frame. The constant terms “C” and “I” are used to represent “COMPLETED” and “INITIATED”, respectively. The predicate CP_i represents a condition predicate specified in the CONDITION slot. The right-hand side of the rule represents the successor of the task in LHS.

The four routing constructs represented using procedural rules are listed in table 1. Tasks are executed sequentially if the execution of one task is followed by the next task (**Rp**$_1$). The parallel routing implies that X and Y can be executed at the same time or in any order if W is completed, and Z can be completed when X and Y have been completed (**Rp**$_2$). On the other hand, conditional routing expresses that X or Y can be executed after W is completed according to the conditions. Z is executed after either X or Y is completed (**Rp**$_3$). Lastly, the iterative routing means that one or more tasks should be repeated until certain condition is satisfied (**Rp**$_4$).

4. Properties of KWM : Soundness

The purpose of workflow model verification is to determine whether the model represents target workflow correctly. There exist some verification techniques for workflow based on Petri-net (Hofstede et al., 1998; Adam et al.,1998; Van der Aalst, 1998). The techniques are limited to the verification of routes such as checking termination of workflow or occurrence of dangling tasks. The rule-based approach of KWM allows verification of correct specifications of rules as well as routes of workflow model. To ensure a KWM represents a workflow correctly, it should satisfy a specific property, soundness.

Definition 5 (Sound) A KWM, *W={E, Rl, Ru}*, is sound if and only if it satisfies the following properties:

(i) *Terminality* : The set *Ru* assures termination of all instances of a workflow.

(ii) *Task Completeness* : The termination of a workflow instance assures termination of all task instances

composing the workflow instance.

(iii) *Compactness* : The set *Ru* assures not occurring redundant rules.

(iv) *Routing Consistency* : There are no conflicting rules in the set *Ru.*

(v) *Referential Integrity* : There does not exist illegal reference in *W.*

The first four properties guarantee the soundness apart from certain anomalies in the set of rule frames. Table 2 summarizes the anomalies that violate the soundness of KWM. The verification of the first four properties can be performed by detecting the anomalies. The occurrence of non-termination of a workflow can happen in three cases; (1) if the inference engine enters a loop in the course of chaining procedural rules (occurrence of circularity) (2) if there are missing rules, and (3) if there are missing values. A dangling task is the task without defined predecessor or successor. The occurrence of dangling tasks can cause an anomaly that the tasks are not completed even though the workflow instance is

completed. The compactness property can be violated if there are redundant rules. The occurrence of redundancy means that some rules or literals in a rule can be removed without affecting the soundness of a KWM. A rule is redundant if it is subsumed or duplicated with other rules. A subsumed rule is that the antecedents of the rule consist of a subset of the antecedents of other rule that has the same consequents with the subsumed rule. If two rules have the same antecedents and consequents, the rules are duplicated.

Table 2. Anomalies that violate soundness of KWM.

Rule Set	Anomaly Explanation
Rp1 = {TS(Initiate, C) ⇒ TS(X,I), TS(X, C) ⇒ TS(Y, I), TS(Y, C) ⇒ TS(X, I), TS(Y, C) ⇒ TS(Terminate, I) }	The set *Rp1* has circularity so that a workflow instance enters a loop between task X and Y.
Rp2 = { TS(Initiate, C) ⇒ TS(X, I), TS(Y, C) ⇒ TS(Z, I), TS(Z, C) ⇒ TS(Terminate, I) }	The sub set *Rp2* is missing a rule that connects task X and task Y. In the case, the workflow instances can not progress after task X is completed.

Rp3 = { TS(Initiate, C) ⇒ TS(X, I), TS(X,C) ∧ LARGER(v,10)⇒ TS(Y,I), TS(X,C) ∧ SMALLER(v, 5)⇒ TS(Z, I) }	If a workflow instance that binds the variable with a value between 5 and 10 is created, the workflow instance becomes dead.
Rp4 = {TS(Initiate, C) ⇒ TS(X, I), TS(X, C) ⇒ TS(Y, I),TS(X, C) ⇒ TS(Z, I), TS(Y, C) ⇒ TS(Terminate, I) }	The task Z does not affect the route of workflow instances. The task Z should be connected to the task Terminate.
Rp5 = { TS(X,C)∧COND1(x)∧COND2(y) ⇒ TS(Y, I), TS(X, C) ∧ COND1(x) ⇒ TS(Y, I), COND1(x) ∧ TS(X, C) ⇒ TS(Y, I)}	In the set *Rp5*, the first rule is subsumed by the second rule, and the second rule is duplicated with the third rule.
Rp6 = { TS(X, C) ∧ LARGER(x, 5) ⇒ TS(Y, I), TS(X, C) ∧ SMALLER(x, 10) ⇒ TS(Z, I)}	The first two rules in the set *Rp6* may infer conflicting hypotheses when a workflow instance binds the variable x with a value between 5 and 10.

The meaning of referential integrity is twofold. First, it restricts the participants to relationships in KWM to be valid entities. That is, if an entity instance that participates in a relationship is removed, the relationship instance should also be removed. Secondly, the referential integrity prevents illegal constraints which constrain the state of non-existent entities or relationships. The rule frames constrain their activation time using the state of entities or relationships in the CONDITION slot. If the condition is defined on the state of non-existing objects, the referential integrity is violated.

For instance, the algorithm in Figure 4 determines whether there exists any missing value in a set of procedural-rule frames. The occurrence of missing values in a procedural rule set means that some parts of domain of an object, which is cartesian product of domains of the object's slots, are not used for defining routing rules after the completion of a task. To check missing values in a set of procedural-rule frames, following steps are followed. At first, for each task *t* in the task set *T*, all the procedural-rule

ALGORITHM 1 (CHECKING MISSING VALUES)

Given a set of procedural-rule frames Rp,

for each task t ∈ T,

Rp(t) = {pr ∈ Rp | pr.PRE_TASK = t}

let O(t) = { o ∈ O | o is restricted in pr.CONDITION and pr ∈ Rp(t)}

for each o ∈ O(t),

check $\vee_{\text{for all } pr \in Rp(t)}$ *pr.CONDITION|o != dom(o) where pr.CONDITION|o is a projected condition of pr.CONDITION,* which is restricted as a condition of object o.

Figure 4. Algorithm for checking missing values

frames that have task *t* as the value of the PRE_TASK slot are extracted. Secondly, all the objects that are used to define conditions in the CONDITION slot of the procedural-rule frames extracted in the first step are selected. Lastly, for each object in the second step, the union of restricted domains of the object that are determined by conditions of procedural-rule frames is calculated. If the union of restricted domain of the object is equal to the domain of the object, there is no missing value. Otherwise, the procedural-rule frames in the first step have missing values for the restriction of the object.

5. Propagation of Change Effect in KWM

Propagation rules against each of the changes are defined using dependencies between modeling constructs to assure the soundness of a KWM. Predicates that represent dependencies among frames of KWM are listed in Table 3. Three types of predicates are considered. The first one is predicate that represents dependencies between entity frames. The relationships that are explained in section 3.1 are transformed into predicates that represent dependencies between entity frames. The second one is predicate that represents dependency between rule frames. Three predicates are considered; 'XOR-firing', 'AND-firing', and 'Substitute'. Only one of the rule frames that are used as arguments of the predicate

Table 3. The predicates that represent dependencies among frames of KWM.

Type	Predicate	Meaning
Entity vs. Entity	IS_A(o1, o2)	Object *o1* inherits from object *o2*
	SUBPART_OF(o1, o2)	Object *o1* is subpart of object2
	works_for(a, u, p)	Actor *a* work for organizational unit *u* with position p
	used_at(re, t)	Resource *re* is used at task *t*
	responsible_for(ro, t)	Role *ro* is responsible for task *t*
	...	…
Rule vs. Rule	XOR-firing(r_1, r_2,…r_n)	One of the rules r_1 , r_2, …, r_n can be fired
	AND-firing(r_1, r_2,…r_n)	All of the rules r_1 , r_2, …, r_n should be fired
	Substitute(r_1, r_2, rm)	Metarule *rm* substitute a set of rules r_1 with a set of rule r_2
Rule vs. Entity	Precedence(t1, t2, rp)	Task t1 precedes t2 with procedural rule rp
	Role-charge(ro, rr)	A responsibility-rule frame *rr* finds actors who are in charge of role *ro*.

'XOR- firing' can be fired. On the other hand, the rule frames that are used as arguments of the predicate 'AND-firing' should be fired concurrently. The predicate 'Substitute' represents the substitution relationship between normal rules and special rules that are represented by a metarule frame. The last type of predicate represents dependencies between entity and rule frames. The predicates 'Precedence' and 'Role-charge' correspond to the type. The predicate 'Precedence' is derived from procedural-rule frame. It represents dependencies between ordered tasks and a procedural rule that define the order. On the other hand, the predicate 'Role-charge' is derived from responsibility-rule frame. It represents dependencies among a role, charged actors, and a responsibility-rule frame that define the mapping relationship.

The algorithm for deriving XOR-firing dependencies between procedural-rule frames is to find a set of rule frames that exclusively constrain on the domain of the same objects. The exclusive procedural-rule frames can be found from conditional routing constructs. The algorithm in Figure 5 constructs a set

(*Rp(t1)*) of procedural-rule frames that should be checked after completion of a task. For each procedural-rule frame in the set *Rp(t1)*, the rule frame is added to a pseudo-exclusive rule set (*XOR(pr1)*). The other rule frames in the set *Rp(pr1)* that constrain the same objects with the procedural-rule frame are, then,

ALGORITHM 2 (FINDING EXCLUSIVE PROCEDURAL-RULE FRAMES)

Given a set of procedural-rule frames Rp,
For each $t \in T$ where T is set of tasks,
set Rp(t1) = {pr∈Rp| pr.PRE_TASK = t}
for each pr1∈Rp(t),
set O(pr1) = {o | o is an object whose domain is restricted in pr1.CONDITION}
set Rp(pr1) = {pr | pr∈Rp(t1), O(pr)=O(pr1), where O(pr) is defined as similar with O(pr1)}
set XOR(pr1) = {pr1}
for each pr∈Rp(pr1),
if pr ∧ ($\wedge_{pri \in XOR(pr1)}$pri.CONDITION) = ∅
pr∈*XOR(pr1)*
if ($\vee_{pri \in XOR(pr1)}pr_i$.CONDITION = $X_{oi \in O(pr1)}dom(O_i)$ where X means cartesian product)
exit
if $\vee_{pri \in XOR(pr1)}pr_i$.CONDITION = $X_{oi \in O(pr1)}dom(O_i)$
Add predicate XOR-firing(pr1, pr2, …, prn) for all pr1, pr2, … , prn ∈ XOR(pr1)

Figure 5. Algorithm for finding exclusive procedural-rule frames.

successively compared to check whether their intersection of constrained domains of the objects is null or not. If the intersection is null, the procedural-rule frames are added to the set pseudo-exclusive rule set. If the comparison is finished for all other rule frames, the union of the constrained domain of rule frames in pseudo-exclusive rule set is calculated. If the union is the same with the entire domain of the objects, then the set of procedural-rule frames constitutes an XOR-firing dependency.

Using the predicates in Table 3, change propagation scope is identified, and proper update on the

affected frames by the change is performed. The change propagation rules, then, are used to automatically modify frames of KWM or notify model builder the anomalies caused by the changes. For instance, some propagation rules against deletion operation on KWM frames are listed in table 4. The propagation rules can trigger other rules, and the propagation chain establishes change propagation scope for a change on a frame.

Table 4. Example of change propagation rules on the deletion operation.

Rule No.	Dependency Predicates	rule specification
		Meaning
5-1	*Responsible_for(ro, t)*	DELETED(t) ⇒ DELETE(ro)
		If a task *t* is deleted, then charged role *ro* can be deleted
5-2	*Precedence(t1, t2, rp)*	DELETED(t1) ⇒ DELETE(rp)
		If a task *t1* is deleted, then procedural rules that have the task as PRE-TASK can be deleted
5-3	*Precedence(t1, t2, rp)*	DELETED(t2) ∧ EQUAL(t2, rp.NEXT_TASK) ⇒ DELETE(rp)
		If a task set *t2* is deleted, then procedural rules that have the set as a whole of NEXT-TASK can be deleted
5-4	*XOR-firing(r1, r2)*	DELETED(r1) ⇒ UPDATE-SLOT(r2.CONDITION)
		If a rule *r1* is deleted, then other rules *r2* that are connected with exclusive OR relationship should be updated
5-5	*Role-charge(ro, rr)*	DELETED(ro) ⇒ DELETE(rr)
		If a role is deleted, then responsibility rules that connects the role with actors can be deleted
5-6	*Substitute(r1, r2, rm)*	DELETED(rm) ⇒ DELETE(r2)
		If a metarule *rm* is deleted, then the rule set *r2* that is defined for the exceptional situation can be deleted

6. An Illustrative Example

The KWM is applied to the business trip approval process at a university (KAIST: Korea Advanced

Institute of Science and Technology) in Korea which has implemented BPR. The overall flow of the "AS-IS" business trip approval process is depicted in Figure 6, where each circle represents a task and a directed arc represents transition of a workflow instance. In a circle, task name and the role that is in charge of the task are specified. Each arc is attached with corresponding procedural rule, and tasks are attached with responsibility rules. The goal of the process is to deal with business trip requests and grants travel allowance according to the organizational rules. The trip applicants can be all members of the university, professors, students, employees, or researchers who work for affiliated research centers.

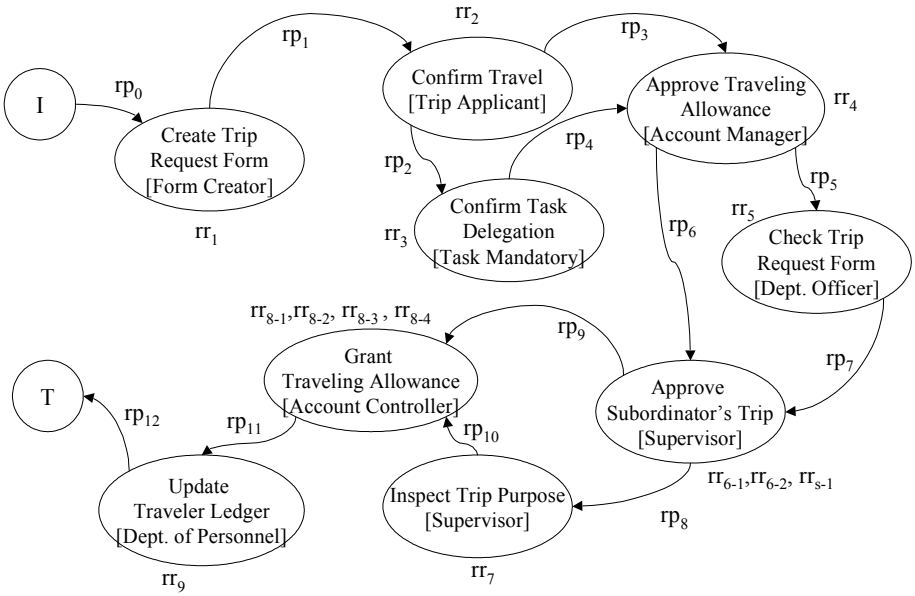

Figure 6. Business trip approval process at KAIST (AS-IS)

6.1 Modeling example workflow using KWM

Figure 7 shows a part of KWM frames for representing the business trip approval process. The rp8, an instance of procedural-rule frame, conditionally routes workflow instance from the task "Approve Subordinator's Trip" to "Inspect Trip Purpose". If trip duration that is specified in the 'duration' slot of

the “Trip Request Form” frame exceeds 6 days, the rule frame is fired.

The frame rr6-1 is an instance of responsibility-rule frame which finds supervisors of the travelers. It represents that the supervisor of a trip applicant is the manager of the department where the trip applicant belongs. If a trip applicant is a manager of a department, however, his/her supervisor is the manager of the trip applicant’s next super department (rr6-2). On the other hand, the account controller of a trip account is determined according to the type of trip applicant. If the trip applicant is a student or professor without any administrative position, the account controller is the one who works for the academic & student services department (rr8-1 and rr8-2). Otherwise, the account controller is determined according

```
FRAME rp8
  DESCRIPTION : “If a supervisor approves the trip request and the trip duration
                    exceeds 6 days, an auditor should inspect the trip purpose.”
  PRE_TASK : Approve_Subordinator’s_Trip;
  PRE_TASK_STATE : “Approved”;
  NEXT_TASK : Inspect_Trip_Purpose;
  CONDITION : (Trip_Request_Form (duration ?dur)) (test (>= ?dur 7)) ;
END_FRAME

FRAME rr6-1
  DESCRIPTION : “The traveler’s supervisor is one who works for the department with
                  manager position which the traveler belongs to”;
  ROLE: Supervisor;
  ACTOR: WorkFor.actor_id;
  CONDITION: (Traveler (Department ?dept-id))
                (Department (dept_id ?dept-id) (mnger_pos ?m-pos))
                (WorkFor (dept_id ?dept-id)(actor_id ?supervisor_id)(position ?m-pos));
END-FRAME

FRAME rm1
  DESCRIPTION : “If a traveler is a director of an affiliated research institute,
        the supervisor is the vice President although his formal supervisor is the President.”;
  SOURCE_RULE: rr6-1;
  TARGET_RULE: rrs-1;
  CONDITION: (Traveler (T_Id ?t-id))
                 (WorkFor (actor_id ?t-id)(position “director-of-affiliated-research-institute”));
END-FRAME
```

Figure 7. Specification examples of procedural-rule frames

to the type of the account from which the traveling allowance is granted. If the traveling allowance is granted from the research project account, the account controller is the one who works for the research management department (rr8-3). In other cases, the account controller is the one who works for the

finance department (rr8-4).

Two exceptional rules exist in the example workflow. At first, if a trip applicant is a director of an affiliated research institute, the vice President of KAIST becomes the applicant's supervisor although the formal supervisor of the research institute is the President (rm1). This exceptional rule existed temporarily to reduce the workload of the President. Secondly, if trip applicant is an employee who is delegated to another department, then the sequence between tasks "Approve Traveling Allowance" and "Approve Subordinator's Trip" is reversed (rm2). In Figure 7, metarule frame rm1 handles an exceptional situation for the trip of a director of an affiliated research institute. The responsibility-rule frame rr-s1 represents mapping relationship between an actor and the deterministic role 'Vice_President'.

6.2 Change Propagation

The business trip approval process in Figure 6 has been changed as a result of BPR project at KAIST. At first, the task "Check Trip Request Form" executed by a department officer is going to be removed from the process because computerized form processing system automatically checks the correctness of the form (Event1). Secondly, the task "Grant Traveling Allowance" is going to be executed only in the finance department and the academic & student services department (deletion of rule rr8-3), and the mapping conditions of other rule frames that are related with XOR-firing dependency should be changed (Event2). Lastly, the exceptional rule for delegated employees is going to be removed, and the workflow instances for the trip of delegated employees should be processed as other normal instances (Event3). Figure 8 shows the change propagation chains using the rules defined in Table 3. The change propagation chains are used to notify workflow modeler the frames that should be updated.

6.5 Advantages of KWM

Applied to business processes at KAIST, KWM has been proved to be useful to automate the business

- For Event1 = DELETED(Check_Trip_Request_Form),

  (1) DELETED(Check_Trip_Request_Form) ⇒ DELETED(rp5) ∧ DELETED(rp7)

  // By [Rule 5-3] and [Rule 5-2]

  DELETED(rp5) ⇒ UPDATE-SLOT(rp6.CONDITION) // By [Rule 5-4]

  // In this case, condition spec. of rp6 should be removed

  (2) DELETED(Check_Trip_Request_Form) ⇒ DELETED(Department_Officer) // By [Rule 5-1]

  DELETED(Department_Officer) ⇒ DELETE(rr5) // By [Rule 5-5]

  // rr5 is a responsibility rule for the role Department Officer

- For Event2 = DELETED(rr8-3),

  DELETED(rr8-3) ⇒ UPDATE-SLOT(rr8-1.CONDITION) ∧ UPDATE-SLOT(rr8-2.CONDITION) ∧ UPDATE-SLOT(rr8-4.CONDITION) // By [Rule 5-4]

- For Event3 = DELETED(rm2),

  DELETED(rm2) ⇒ DELETE(rp-s1, rp-s2, rp-s3, rp-s4, rp-s5) // By [Rule 5-6]

Figure 8. Change propagation chains for the example workflow.

processes in organizations under changing environment. Three main advantages for workflow management are observed. First, the rule-based approach of KWM enables one to represent complex business rules of conditional routing and role/actor mapping under organizational context. It is also appropriate to represent exceptional rules that are applied to special workflow instances. KAIST consists of heterogeneous actor types such as student, professor, employee, and researchers. KWM is applicable for modeling the business rules that are changing according to the user type. Furthermore, with the rule expression power, KWM can be used as a computerized rulebook that reflects the organization's contextual knowledge. Agostini (1996) and Kirn (1994) addressed the importance of modeling organization's contextual knowledge in cooperative information systems. Organizational context knowledge serves workflow participants as a virtual expert in workspaces. Secondly, workflows that are executed by complex business rules are apt to be specified incorrectly. Representing the rules in workflow model can easily result in redundant, inconsistent, and incomplete rules. Providing model verification techniques for KWM increases the correctness of workflow specifications. Lastly, increasing the

adaptability of model is the main advantage of WFMS. KWM is an adaptive model in that organizational changes on the organizational structure and business rules as well as procedures are reflected through firing the change propagation rules. Furthermore, providing metarule frames for representing exceptional rules mitigates the modification burden of dynamic workflow model. All the exceptions that are expected before execution of workflows can be explicitly represented using the metarule frames.

7. Conclusion

This paper presents a knowledge-based approach to increase the adaptability of WFMS against organizational changes. KWM is useful for agile organizations that frequently change their business processes under turbulent organizational environment. Particularly, it has been designed to be adaptable to the changes on organizational structure, business rules and procedures represented as task sequences. KWM has the following features;

First, expressive power of workflow model is improved. The rule-based approach of KWM enables representing complex business rules such as routing works to actors and assigning tasks to actors according to responsibilities of organizational roles. Furthermore, exceptional rules that are applied to special workflow instances are also represented explicitly.

Secondly, verification technique for KWM has been suggested to check inconsistent, incomplete, and redundant rules as well as non-termination of workflow.

Lastly, the dependencies between frames of KWM are maintained via predicates, and change propagation rules have been defined using the dependencies. The change propagation rules assure the correctness of KWM against the changes on the organizational structure, business rules and procedures.

K-WFMS (Knowledge-based Workflow Management System) has been fully implemented using CLIPS and integrated as a component of a campus-wide information system called Intelligent Campus at KAIST (Park et al, 1995 & 1996). K-WFMS is integrated with other application information systems for executing tasks in business processes. With the successful real application, KWM has proven to be a useful framework to implement the fully automated workflow under the agile environment.

References

Adam, N.R., Atluri, V., and Huang, W. (1998). Modeling and analysis of workflows using Petri Nets, *Journal of Intelligent Information System,* 10, 131-158.

Agostini, A., Michelis, G.D., Grasso, M.A., Prinz, W., and Syri, A. (1996). Contexts, work processes, and workspaces, *Computer Supported Cooperative Work, 5, 223-250.*

Bogia, D. P. and Kaplan, S.M. (1995). Flexibility and control for dynamic workflows in the worlds environment. *Proc. of the Conference on Organizational Computing Systems* (ACM 1995).

Bose, Ranjit (1996). Intelligent agents framework for development knowledge-based decision support system for collaborative organizational processes, *Expert System With Applications,*11(3), 247-261.

Casati, F., Ceri, S., Pernici, B., and Pozzi, G. (1996). Deriving active rules for workflow enactment, In *Proceedings of DEXA 96,* Zurich(CH), 94-110.

Casati, F., Ceri, S., Pernici, B., and Pozzi, G. (1998). Workflow evolution, *Data and Knowledge Engineering,* 24, 211-238.

Chang, J.W. and Scott, C.T. (1996). Agent-based workflow: TRP Support environment (TSE), *Computer Networks and ISDN Systems.* 28, 1501-1511.

Dellen B., Maurer, F., and Pews, G. (1997). Knowledge-based techniques to increase the flexibility of workflow management, *Data and Knowledge Engineering,* 23, 269-295.

Ellis,C.A. and Nutt, G.J. (1993). Modeling and enactment of workflow systems. *Application and Theory of Petri Nets 1993, 14th International Conference Proceedings*, Chicago, Illinois, USA, 1-16.

Flores, F., Graves, M., Hartfield, B., and Winograd, T. (1988). Computer systems and the design of organizational interaction, *ACM Transactions on Office Information Systems,* 6 (2), 153-172.

Goldman, S. L., Nagel, R. N., and Preiss, K. (1995). *Agile Competitors and Virtual Organizations: Strategies for Enriching the Customers*, Van Nostrand Reinhold, 1995.

Gruhn, Volker(1995). Business process modeling and workflow management, *International Journal of Cooperative Information Systems*. 4(2&3), 145–164.

Hofstede, A.H.M., Orlowska, M.E., and Rajapakse, J. (1998). Verification problems in conceptual workflow specifications, *Data and Knowledge Engineering,* 24, 239-256.

Jennings, N.R., et al. (1996). Agent based business process management, *International Journal of Cooperative Information System*. 5 (2), 105-130.

Kappel, G., Lang, P., Rausch-Schott, S., Retschitzegger, W. (1995). Workflow management based on objects, rules, and roles, *IEEE Bulletin of the Technical Committee on Data Engineering*, 18(1).

Kaplan, S., Tolone, W., Bogia, D., and Bignoli, C. (1992). Flexible, active support for collaborative work with ConversationBuilder. *Proc. of the Conference on Computer Supported Cooperative Work (Toronto, Canada) (378-385)*. NY, ACM/SIGCHI and SIGOIS.

Kirn, S. (1994). Supporting human experts' collaborative work: Modeling organizational context knowledge in cooperative information system. In John H. Connolly, and Ernest A. Edmonds (Eds.), *CSCW and Artificial Intelligence*. Springer-Verlag. 127-139.

Marshak,R.T. (1994). An overview of workflow structure. *Proc. of Workflow(13-42).* San Jose, USA: Future Strategies Inc.

Medina-Mora, R., Winograd, T., Flores, R., and Flores, F. (1992). The action workflow approach to workflow management technology. *Proc. of Computer Supportive Cooperative Work (281-288).*Toronto, Canada.

Michelis, G. D. and Grasso, M. A. (1994). Situating conversations within the language/action perspective: The Milan conversation model. *Proc. of Computer Supported Cooperative Work (89-100),* Chapel Hill, NC, USA.

Park, J.Y. and Park, S.J. (1998). IBPM: An integrated systems model for business process reengineering, *Systems Engineering,* forthcoming.

Park, S.J. (1996). Development of future management information system for intelligent campus, *Project Report*, KAIST, Korea.

Park, S.J., Kim, J.W., Lee, H.B., Cho, K.H., and Kim, G.J. (1995). Open Workflow Automation Systme in Client/Server Environment. *Proc. of the Korea Society of Management Information Systems (149-158).* Seoul, South Korea.

Peterson, J. L. (1981). *Petri Net Theory and the Modeling of Systems*. Prentice-Hall Inc.

Preece, A. D., Batarekh, A., and Shinghal, R., (1992), Verifying rule-based systems, *Knowledge Engineering Review,* 7(2), 115-141.

Reichert, M. and Dadam, P. (1998). $ADEPT_{flex}$-Supporting dynamic changes of workflows without losing control, *Journal of Intelligent Information Systems,* 10, 93-129.

van der Aalst, (1998). The application of Petri nets to workflow management. *The Journal of Circuits, Systems and Computers,* forthcoming.

WfMC (1996). *Workflow management coalition terminology and glossary (WfMC-TC-1011)*. Technical Report, Workflow Management Coalition, Brussels.

Winograd, T. (1987). A language/action perspective on the design of cooperative work, *Human-Computer Interaction,* 3 (1), 3-30.

Wolf, M. and Reimer, U. (1996). *Proc. of the International Conference on Practical Aspects of Knowledge Management (PAKM '96), Workshop on Adaptive Workflow,* Basel, Switzerland, Oct.