

CERIAS Tech Report 2005-73

A FRAMEWORK FOR MANAGEMENT OF SECURE AND ADAPTIVE WORKFLOWS

by Basit Shafiq, Arjmand Samuel, Elisa Bertino, and Arif Ghafoor

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

A Framework for Management of Secure and Adaptive Workflows

Basit Shafiq, Arjmand Samuel, Elisa Bertino, and Arif Ghafour
Purdue University
{shafiq, amsamuel}@purdue.edu

Abstract

In this paper, we propose a framework for secure composition and management of time based work flows. The proposed framework allows communication and sharing of information among predefined or ad hoc team of users collaborating with each other in the time critical workflow applications. A key requirement for such applications is to provide the right data to the right person at the right time. In addition, the workflow needs to be adapted if a subtask of a workflow cannot be executed within the due time. The proposed framework supports GTRBAC based workflow specification and allows dynamic adaptation of workflow instances depending on the execution status of workflow tasks and environmental context. Adaptations in a workflow may include rescheduling of component tasks, reassignment of users to the scheduled tasks, or abortion of component tasks that cannot be completed under the current system state. We propose an integer programming based approach for finding the best possible adaptation according to the pre-defined optimality criterion.

1. Introduction

Distributed workflow based systems are widely used in various application domains including e-commerce, digital government, healthcare, power systems, air traffic control, manufacturing and many others. Workflows in these application domains are not restricted to the administrative boundaries of a single organization and may require inter-organization information and resource sharing for the execution of the tasks comprising the workflow [3, 9, 12]. The tasks in a workflow need to be performed in a certain order and often times are subject to temporal constraints and dependencies [8, 7, 5]. Temporal constraints may be in the form of strict deadlines that must be met for correct execution of the workflow application. A key requirement for such time-based workflow applications is to provide the right data to the right person at the right time. This requirement motivates for dynamic adaptations of workflows for dealing with changing

environmental conditions and exceptions. For example, in a given workflow instance a pre-assigned user for a given task may not be able to complete the task within due time because of excessive load. Therefore, the task needs to be assigned to a new user for timely completion of the workflow instance.

For any workflow instance, it is essential that the underlying workflow tasks are executed by authorized users only. Therefore, appropriate access control mechanisms need to be employed to meet this requirement. Traditional access control models such as Discretionary and Mandatory Access Control (DAC and MAC) lack the capability to capture the time-based dependencies of workflow applications. The recently proposed Generalized Temporal Role-Based Access Control (GTRBAC) [10] model provides a suitable approach for specification of security and access control requirements of time-based workflow applications. Role-based authorization considerably reduces the management overhead in terms of policy specification. The most distinguishing feature of GTRBAC is the support for temporal constraints which is essential for modeling the real-time dependencies. Additionally, GTRBAC allows specification of separation of duties (SoD), cardinality, and event dependency constraints that are required in many workflow based applications.

In [11], we presented a framework for dynamic composition and management of time-based workflows. The framework supports GTRBAC based workflow specification and allows dynamic adaptation of active workflows depending on the execution status of workflow tasks and environmental context. There can be several possibilities for workflow adaptation with different trade-offs. In this paper, we extend the proposed framework to incorporate a workflow management component that performs workflow adaptation in an optimal manner under the given security constraints and environmental context. In particular, we propose a mixed integer programming (MIP) based technique for finding the best possible workflow adaptation according to the pre-defined optimality criterion. The proposed technique is generic and can be tuned to a variety of optimality measures including minimization of task execution delays,

minimization of user-task reassignments, and minimization of constraint relaxation.

The remainder of the paper is organized as follows. In Section 2, we present an overview of the GTRBAC model. In Section 3 we briefly discuss the software architecture of the extended framework for composition and management of adaptive workflows. The proposed technique for optimal workflow adaptation is described in Section 4. Section 5 provides an illustrative example demonstrating the usability of the proposed technique. Section 6 presents related work and Section 7 concludes the paper.

2. Overview of GTRBAC Model

In this section, we briefly overview the GTRBAC model used to specify the access control policies and work flow semantics. GTRBAC is a temporal extension of the role-based access control (RBAC) model proposed by Sandhu *et. al.* in [15]. RBAC consists of the following four basic components: a set of users *Users*, a set of roles *Roles*, a set of permissions *Permissions*, and a set of sessions *Sessions*. A user is a human being or a process within a system. A role is a collection of permissions associated with a certain job function within an organization. A permission is an access mode that can be exercised on a particular object or resource in the system. A session relates a user to possibly many roles and allows the user to access all permissions associated with such roles.

One of the important aspects of access control is that of time constrained accesses to limit resource use. Such constraints are essential for controlling time sensitive activities that may be present in various applications such as workflow management systems (WFMSs) [10, 4] and real-time active databases [16]. To address general time-based access control needs, Joshi *et. al.* [10] have proposed a Generalized Temporal RBAC (GTRBAC) model. A key aspect of the GTRBAC model is the notion of states of a role. In GTRBAC, a role can assume one of the three states: *disabled*, *enabled*, and *active*. A role is *enabled* if a user can acquire the permissions assigned to it. An *enabled* role becomes active when a user acquires the permissions assigned to the role in a session. By contrast, a *disabled role* cannot be activated by any user. Therefore, constraints on enabling of roles specify when roles can actually be assumed by users. The GTRBAC model provides a complete framework for specification of temporal constraints on all events related to user-role and role-permission assignment, role enabling/disabling, and user-role activation.

Table 1: GTRBAC model specifications used in this paper

Constraint Categories	Expression	Explanation
Duration Constraints	$([t'_f, t'_s] \text{ enable } r)$	Role r enabled from t'_s to t'_f
	$([t^\square_f, t^\square_s], [d_{min}, d_{max}] \text{ execute } \square)$	Task \square activation, starting time t^\square_s and finish time t^\square_f such that $\max \square (t^\square_f - t^\square_s) \square \min$
Assignment Constraint	$\text{assign}(u_i, r)$ for any user i	Assign user u_i to role r
Role Task Assignment	$\text{assign}(r, \square)$	Role r is assigned to task \square
User task activation	$\text{execute}(u_i, \square)$ for any user i	User u_i executes the task \square
Hierarchy	$r_i \square r_j, r_i, r_j \square \text{Roles } r$	r_i inherits the permissions of r_j
Role-specific SoD	$RSoD(r_1, r_2)$	No user can assume role r_1 and role r_2 in the same workflow instance
Task-specific SoD	$TSoD(\square_1, \square_2)$	No user can execute the tasks \square_1 and \square_2 in the same workflow instance
User-specific SoD (role-based)	$USoD(u_1, u_2, r)$	User u_1 and user u_2 are conflicting for role r
User-specific SoD (task-based)	$USoD(u_1, u_2, \square)$	User u_1 and user u_2 are conflicting for task \square

Tables 1 summarizes the type and formal expressions of GTRBAC constraints considered in this paper. Duration constraints are used to specify durations for which role enabling or user-role activation event is valid. The expression $([t'_f, t'_s] \text{ enable } r)$ implies that role r is enabled from time t'_s to t'_f . $([t^\square_f, t^\square_s], [d_{min}, d_{max}] \text{ execute } \square)$ implies that task \square needs to be executed at time t^\square_s and must finish by time t^\square_f . The minimum execution duration for completion of task \square is d_{min} and maximum duration is d_{max} . Assignment constraints are used to specify the user-role and task-role assignments. The assignment expression $\text{assign}(u_i, r)$ assigns user u_i to role r . Similarly, the role task assignment $(\text{assign}(r, \square))$ denotes the assignment of task \square to role r . The user-task activation expression $\text{execute}(u_i, \square)$ implies that the task \square needs to be executed by user u_i . The role hierarchy constraint $r_i \square r_j$ specifies

that role r_i is senior to role r_j . By virtue of this hierarchy relationship, r_i inherits all the permissions of role r_j . The role separation of duty constraint $RSoD(r_1, r_2)$ implies that no user can assume role r_1 and role r_2 concurrently. Similarly, user separation of duty constraint $USoD(u_1, u_2, r)$ implies that user u_1 and user u_2 are conflicting for role r . A detailed explanation of these constraints can be found in [17].

3. Software Architecture

Figure 1 depicts the software architecture for dynamic workflow composition and management. The architecture consists of three key components, including, workflow composition module, workflow management module, and access control module.

3.1. Work Flow Composition Module

The workflow composition module (WCM) provides an authoring tool for specification of workflow tasks and the interdependencies between these tasks. In addition, other dynamic constraints including separation of duties and task execution cardinality constraints can be specified for the underlying workflow tasks. GTRBAC formalism is used to specify all the workflow constraints. The consistency analyzer component in WCM is responsible for checking the consistency and the correctness of the composed workflow in terms of task dependencies, deadlines, and constraint conflicts.

3.2. Access Control Module

The access control module (ACM) is responsible for determining the authorization of users for execution of workflow tasks. The authorizations of users are determined based on their assigned roles. This assignment may be pre-defined in the access control policy or may be performed dynamically based on users' credentials and the context parameters. The credentials facilitate in authentication process and are supplied by the user upon the request of ACM. In addition, user credentials also help in determining the qualification and skill level of the user. The context parameters may be specific to the user, such as user location, time of access, and the current resource capacity. Additionally, environmental context such as system load and execution status is also considered in determining the user assignment. The user-specific context is extracted from the information supplied by the user at the time of access request and the environmental context is provided by the state monitoring module. In a distributed workflow environment, users executing workflow tasks may belong to different organizations or

administrative domains. To enable collaboration in such workflow environment, dynamic role mapping is needed that defines the relationship between the roles assumed by the users in their own organizations and the roles assigned to users for workflow execution. The role mapping/assignment component in ACM is also responsible for creating such mapping.

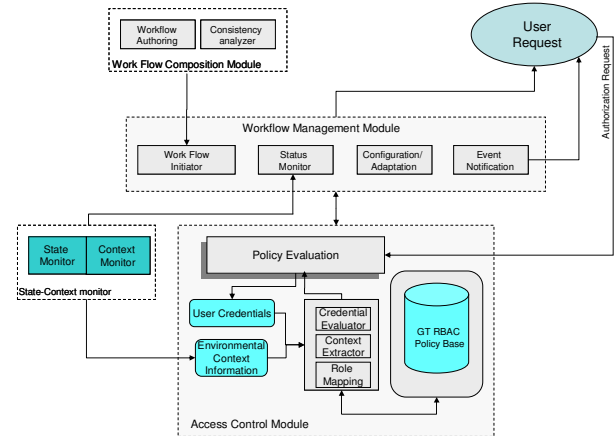


Figure 1. Software architecture for dynamic workflow composition and management

3.3. Work Flow Management Module

The workflow management module (WMM) is the most important component of the proposed architecture. It consists of the following key sub-components: workflow invocation component, execution status monitor, event notification component, and workflow reconfiguration/adaptation component. The workflow invocation component in WMM is responsible for instantiation of a workflow upon the request of an authorized user. The authorization of workflow instantiation request is determined by ACM. ACM also performs user to role assignment for the different tasks of the instantiated workflow. Upon receiving the authorization approval the workflow invocation component creates an instance of the workflow with the user to role bindings for the associated workflow tasks. After the instantiation of the workflow, the corresponding users are notified for the execution of the tasks assigned to them. The event notification component is responsible for sending out such notification messages to the corresponding users at the appropriate time.

The execution of an instantiated workflow may not proceed as planned in the invocation phase. Changes in the environmental or user context or the occurrence of certain unpredictable events may block the execution of some tasks in the workflow. Consequently, the workflow needs to be reconfigured for execution of blocked tasks. The workflow adaptation/reconfiguration component is

responsible for such adaptations. Workflow adaptation is triggered by the status monitor component which continuously checks the execution status of all active and pending tasks. Adaptations in a workflow may include rescheduling of certain workflow tasks, relaxation of policy constraints, reassignment of users to the scheduled tasks based on their availability, authorization, and skill level, or abortion of certain tasks that cannot be completed under the current system state. Depending on the execution status of workflow tasks and environmental context, several adaptation possibilities may exist for an active workflow. The adaptation/reconfiguration component needs to find an adaptation that optimizes the overall performance under the given constraints.

Table 2. MIP variables and their explanation

Variable	Corresponding Vector	Type	Interpretation
t_s^\square	\mathbf{t}^\square	R ⁺	Start time of task \square
t_f^\square	\mathbf{t}^\square	R ⁺	Finish time of task \square
t_s^r	\mathbf{t}^r	R ⁺	Time at which role r enters into <i>enable</i> state from <i>disable</i> state
t_f^r	\mathbf{t}^r	R ⁺	Time at which role r enters into <i>disable</i> state from <i>enable</i> state
i^\square	\mathbf{I}^\square	Binary	Variable indicating execution feasibility of task \square $i^\square=1$ if \square can be executed; otherwise, $i^\square=0$
u_i^\square	\mathbf{u}^\square	Binary	Variable indicating execution of task \square by user u_i . $u_i^\square=1$, if u_i executes \square ; otherwise, $u_i^\square=0$
u_i^r	\mathbf{u}^r	Binary	Variable indicating activation/assumption of role r by user u_i . $u_i^r=1$, if u_i assumes r ; otherwise, $u_i^r=0$
c_k	\mathbf{c}	Binary	Constraint variable. $c_k = 0$ if constraint c_k is relaxed in the workflow instance; otherwise $c_k = 1$.
s_r^\square	\mathbf{s}^\square	Binary	$s_r^\square = 1$ if $t_s^r \leq t_s^\square$, i.e., r is enabled prior to the start time of \square
f_r^\square	\mathbf{f}^\square	Binary	$f_r^\square = 1$ if $t_s^r \leq t_s^\square$, i.e., \square finishes before r gets disabled
d^\square	\mathbf{d}	R ⁺	Scheduling delay of a task \square with respect to the original workflow specification
\square_r^\square	\square	Binary	Variable indicating that the execution interval of task \square is contained in the enabling interval of role r . $\square_r^\square = s_r^\square f_r^\square$

4. An Approach for Optimal Workflow Adaptation

In this section, we describe the proposed mixed-integer programming (MIP) based approach for workflow adaptation. The proposed approach determines an optimal adaptation for an active workflow instance based on a predefined optimality criterion. Various optimality

measures, such as minimum task execution/scheduling delays, minimum task to user reassignments and minimum constraint relaxations, can be specified. Depending on the application requirements a hybrid of these optimality measures can also be employed.

4.1. MIP Formulation

The workflow adaptation problem can be formulated as the following mixed integer program:

$$\begin{aligned} & \text{Maximize } \mathbf{w}_1^T \mathbf{u}^\square + \mathbf{w}_2^T \mathbf{c} - \mathbf{w}_3^T \mathbf{d} \\ & \text{Subject to } \mathbf{A}(t) [\mathbf{u}^\square \mathbf{u}^r \mathbf{t}^\square \mathbf{t}^r \mathbf{I}^\square \mathbf{c} \ \square \ \mathbf{d}] \leq \mathbf{b}(t) \\ & \square u_i^\square \in \{0, 1\}, u_i^\square = 0 \text{ or } 1, \square u_i^r \in \{0, 1\}, u_i^r = 0 \text{ or } 1, \square c_k \in \{0, 1\}, \\ & \square i^\square \in \{0, 1\}, i^\square = 0 \text{ or } 1, \square \square_r^\square \in \{0, 1\}, \square_r^\square = 0 \text{ or } 1, \\ & \square t_s^\square \text{ and } t_f^\square \in \mathbb{R}^+, t_s^\square \geq 0 \text{ and } t_f^\square \geq 0, \square t_s^r \text{ and } t_f^r \in \mathbb{R}^+, t_s^r \geq 0 \\ & \text{and } t_f^r \geq 0, \\ & \square d_j \in \mathbb{R}^+, d_j \geq 0, \square w \in [\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3], w \geq 0 \end{aligned}$$

where, $w = [w_1 \ w_2 \ w_3]$ is a weight vector specifying the optimality criteria in the objective function. \mathbf{u}^\square is a vector capturing users authorization for task executing in the given workflow instance. The binary variable $u_i^\square \in \{0, 1\}$ takes a value of one if user u_i executes task \square otherwise u_i^\square takes a value of zero. Similarly, the vector \mathbf{u}^r captures users authorization for role activation. If a user u_i is not authorized to activate role r then u_i^r is set to zero. u_i^r takes a value of one if user u_i has activated role r . The elements of the vectors \mathbf{t}^\square specify the start and finish time of the corresponding tasks. Similarly, the elements of vector \mathbf{t}^r defines the enabling intervals of roles in the given workflow instance. \mathbf{I}^\square is a vector of task indicator variables specifying execution feasibility of corresponding tasks in the workflow instance. A variable $i^\square \in \{0, 1\}$ is assigned a value of one if the task \square can be executed in the given workflow instance. \mathbf{c} is a constraint vector whose elements corresponds the workflow or policy constraints including inter-task dependency and SoD constraints. The elements of the vector \square specify the containment relation between the enabling interval of the roles and the execution duration of tasks as explained in Table 2. \mathbf{d} is a delay vector whose elements capture the scheduling delays of task in the reconfigured or adapted workflow with respect to the scheduled time of the respective tasks in the original workflow specification. The elements of matrix \mathbf{A} and vector \mathbf{b} correspond to the coefficients of terms used in the equations/inequalities defining the MIP constraints. The rules for generating MIP constraints from a given workflow instance are presented in Section 4.2. Both \mathbf{A} and \mathbf{b} are functions of time and the execution status of the workflow instance, implying that the MIP constraints for two workflow instances will be different if the adaptation procedures are invoked at different times. The variables

used in the above MIP formulation are explained in Table 2.

4.2. MIP Constraint transformation rules

In the following, we provide rules for generating MIP constraints from a given workflow specification and access control policy.

1. *Temporal constraint on role enabling*: The role enabling constraint ($[t_1, t_2]$ enable r) can be captured in the IP problem as: $t_s^r = t_1$ and $t_f^r = t_2$.

2. *Task Duration constraint*: The task duration constraint ($t_1, [d_{\min}, d_{\max}]$, enable Δ) can be captured in the IP problem as: $t_s^\Delta = t_1$, $t_f^\Delta - t_s^\Delta \square d_{\min} i^\Delta$, and $t_f^\Delta - t_s^\Delta \square d_{\max} i^\Delta$.

3. *Temporal constraint between role enabling and task execution*: Let a task Δ be assigned to role r or to a role junior to r in the role hierarchy. The task Δ can be executed by assuming role r if the execution duration of Δ is contained within the enabling interval of role r , i.e., $\square_r^\Delta = 1$. The following inequalities specify this containment constraint

- $t_s^r - t_s^\Delta \square (1 - s_r^M)M$
- $t_s^\Delta - t_s^r \square s_r^M$
- $t_f^\Delta - t_f^r \square (1 - f_r^M)M$
- $t_f^r - t_f^\Delta \square f_r^M$
- $\square_r^\Delta = s_r^\Delta f_r^\Delta$

where, M is a very large number ($M \gg 1$). For $\square_r^\Delta = 1$, role r need to be enabled prior to the start time of Δ (constraints a and b) and r cannot get disabled before Δ finishes (constraints c and d).

4. *User-task Authorization constraint*: A task can only be executed by authorized users. A user u assigned to role r can execute a task Δ assigned to role r' if $r = r'$ or $r \square r'$. Let U_Δ be the set of all users authorized to execute task Δ and the variable i^Δ be an indicator variable for execution of task Δ . The following inequalities relate the user-task execution variables to the indicator variable i^Δ , implying that only authorized users can execute the task Δ

- $\sum_{u_j \square U_\Delta} u_j^\Delta - i^\Delta \square 0$
- $u_j \square U_\Delta u_j^\Delta - i^\Delta \square 0$

5. *User-role assumption/activation constraint for task execution*: Consider a task Δ assigned to role r' . Let R be a set of roles containing the role r' and its senior roles. i.e., $R = \{r' \mid (r = r') \square (r \square r')\}$. An authorized user u_i can execute task Δ if u_i assumes any role $r \square R$ and r remains in enable state for the entire execution duration of Δ . Formally:

$$\sum_{r \square R} \square_r^\Delta u_i^r \square u_i^\Delta \square 0.$$

Table 3. MIP formulation of constraints








Temporal Dependency constraint ' c_i ' between \square_1 and \square_2	IP Constraints
 $equal(\square_1, \square_2)$	$c_i(t_s^{\square_1} - t_s^{\square_2}) = 0$ and $c_i(t_f^{\square_1} - t_f^{\square_2}) = 0$
 $before(\square_1, \square_2, \Delta)$	$c_i(t_s^{\square_2} - t_f^{\square_1} - \Delta) \square 0$
 $meets(\square_1, \square_2)$	$c_i(t_s^{\square_2} - t_f^{\square_1}) = 0$
 $starts(\square_1, \square_2, \Delta)$	$c_i(t_s^{\square_2} - t_s^{\square_1} - \Delta) \square 0$
 $finishes(\square_1, \square_2, \Delta)$	$c_i(t_f^{\square_2} - t_f^{\square_1} - \Delta) \square 0$
 $during(\square_1, \square_2, \Delta)$	$c_i(t_s^{\square_1} - t_s^{\square_2} - \Delta) \square 0$ and $c_i(t_f^{\square_1} - t_f^{\square_2}) \square 0$
 $overlap(\square_1, \square_2, \Delta)$	$c_i(t_s^{\square_2} - t_s^{\square_1} - \Delta) \square 0$ and $c_i(t_s^{\square_2} - t_f^{\square_1}) \square 0$

Table 4. Translation of SoD constraints to the corresponding MIP constraints

SoD Constraint ' c_i '	Explanation	MIP Constraint
Role-specific	Two conflicting roles r_1 and r_2 cannot be activated by same user in the same workflow instance.	$\square u_i \square USERS, c_i(u_i^{r_1} + u_i^{r_2} - 1) \square 0$
Task-specific	Two conflicting tasks \square_1 and \square_2 cannot be executed by same user in the same workflow instance.	$\square u_i \square USERS, c_i(u_i^{\square_1} + u_i^{\square_2} - 1) \square 0$
User-specific role-based	Let U_r be the set of users conflicting for role r . At most one user from the conflicting user set U_r can activate role r in any given workflow instance.	$c_i \square u_j^r \square 1 \square 0$ $u_j \square U_r$
User-specific task-based	Let U_Δ be the set of users conflicting for task Δ . At most one user from the conflicting user set U_r can execute the task Δ in any given workflow instance.	$c_i \square u_j^\Delta \square 1 \square 0$ $u_j \square U_\Delta$

6. *Inter-task temporal dependencies*: The temporal dependency constraints between two tasks τ_1 and τ_2 can be specified using Allen's temporal relation [18]. These temporal relations and the corresponding MIP constraints are shown in Table 3. In the MIP constraints of Table 3, the binary variable c_i denotes the corresponding dependency constraint. $c_i = 1$ implies that the dependency constraint is corresponding to c_i is preserved in the adapted workflow instance. If the dependency constraint cannot be satisfied then $c_i = 0$.

7. *Separation of duty (SoD) constraints*: There are four types of SoD constraints, namely: role-specific SoD, task-specific SoD, user-specific role-based SoD, and user-specific task-based SoD. These SoD constraints and the corresponding MIP constraints are shown in Table 4. In the MIP constraints of Table 3, the binary variable c_i denotes the corresponding SoD constraint. $c_i = 1$ implies that the respective SoD constraint is preserved in the adapted workflow instance. If the SoD constraint cannot be satisfied then $c_i = 0$.

4.3. Optimality Criteria and Weight Assignment

The rules described in the above section are used to transform a workflow instance and policy specification into MIP constraints. Once all the MIP constraints for a given workflow instance are defined, an optimal adaptation/reconfiguration of the workflow instance can be achieved by solving the resulting optimization problem. The optimality measure is embedded in the objective function of the corresponding MIP problem. In the above MIP formulation, different optimality measures can be defined for workflow adaptation. These optimality measures include minimizing overall task scheduling delays if certain tasks need to be rescheduled for workflow progress, minimizing reassignment of users to the scheduled tasks if the workflow instance cannot proceed to completion with the original user-task assignment, and minimizing relaxation of workflow or policy constraints if such constraints cause a deadlock in the workflow execution.

The elements of the vector \mathbf{d} in the objective function, captures the task scheduling delays of the adapted workflow with respect to the task scheduling times of the original workflow instance. In case the adaptation requirement is to complete the blocked workflow with minimum overall delay without any constraint relaxation, then the objective function only comprises of delay variables. In this case all the task indicator variables and constraint variables are set to one in order to capture the task completion and constraint satisfaction requirements. Note that the optimality measure in this case does not consider keeping the same user-task assignment specified in the original workflow. Reassigning workflow tasks to

new users may not be desirable if such reassignment increases workflow execution cost. In order to minimize such cost, the preferred user-task assignment needs to be specified in the objective function. The decision variable u_i^{τ} in the objective function specifies that user u_i^{τ} is the preferred user for execution of task τ . All the user-task variables in the vector \mathbf{u}^{τ} with non-zero weight coefficients in the objective function represent the preferred users for the respective tasks, and the IP solution that maximizes such objective function minimizes user-task reassignment.

In some cases, certain workflow and policy constraints may cause a deadlock in workflow execution. To resolve such deadlock, either the entire workflow is aborted or some of the conflicting constraints are relaxed for workflow completion. The choice for such deadlock resolution due to conflicting constraints is application dependent. In case the application allows relaxation of some workflow and policy constraints, then the workflow instance need to be adapted with minimum constraint relaxation. In this case, only the constraints variables that can be relaxed are listed in the objective function with non-zero weight. Maximizing the value of such objective function that comprises only constraint variables, amounts to retaining the maximum number of constraints in the reconfigured workflow.

The optimality measure may also be a hybrid of the three objectives discussed above. However, minimizing task-scheduling delays, user-task reassignment, and constraint relaxation may be conflicting goals. In this case, the optimality of the solution is determined based on the priorities of individual decision variables. The weight vector in the objective function specifies such priorities. Depending on the application requirements, the relative priorities among conflicting parameters in the objective function can be determined. For instance, a given workflow application may prefer retaining the original user assignment for a task τ if the scheduling delay for τ in the adapted workflow is less than ten time units. Suppose user u_i was originally assigned to perform task τ . In the objective function, let the variable d_1 denote the scheduling delay for task τ and u_i^{τ} represent u_i 's assignment for execution of task τ . In this case, the weight assigned to the delay variable d_1 is one-tenth of the weight of the decision variable u_i^{τ} .

5. Illustrative Example

In this section, we illustrate the proposed workflow adaptation technique using an example of purchase order processing workflow as shown in Figure 2. The workflow starts with the task of preparing product requirements (PR). These requirements specify the quantity, size,

quality, and other features of the raw materials and component parts, and are prepared by the personnel of Engineering Department. The next task is the finance approval (FA) for requesting quotations from different vendors. The request for quotation (RFQ) is prepared by an employee of the Purchase department. The RFQ document is made available to the various vendors so that they can quote their prices. In Figure 2, the task of receiving vendor quotations from i^{th} vendor is represented as VQ^i . After receiving quotations from all vendors, the quotations are consolidated (CQ) for reviewing. The review of consolidated quotes is performed separately by authorized employees of Engineering Department (RWE), Finance Department (RFA), and by a management executive (RWM). After these reviews, one of the quotes is selected (SQ) for purchasing. In a workflow process, some of the tasks conflict with each other and need to be executed by different users. This constraint is imposed by defining separation of duty constraints between the conflicting tasks. In the workflow process of Figure 2, the following pairs of tasks are conflicting and have a SoD constraint between them: PR-FA, FA-RFQ, RFQ-CQ, and CQ-SQ. The temporal ordering of the tasks and inter-task dependency constraints of the purchase order workflow is shown in Figure 3(a).

The authorizations for executing different tasks of the purchase order workflow are specified in the access control policy of the organization. This access control policy is defined using GTRBAC model with following roles: General Manager Engineering (GME), General Manager Purchase (GMP), Manager Purchase History (MPH), Manager Engineering Quality (MEQ), Audit Clerk (AC), Vendor (V), Purchase Clerk (PC), Manager Finance (MF) and Manager Engineering Development (MED). The role hierarchy, user-role assignment, and role-permission assignment specified in the access control policy of the organization is shown in Figure 3(c). In this figure, the gray boxes represent the roles and the white boxes represent the tasks. The users assigned to a role are shown next to the role. For instance, users u_5 , u_6 , and u_7 are assigned to the role GMP which is senior to the roles PC and AC. Task SQ is assigned to the role GMP, RFQ is assigned to the role PC, and CQ is assigned to the role AC. Since users u_5 , u_6 , and u_7 are assigned to the senior role GMP, they can assume all three roles GMP, PC and AC and can execute all workflow tasks assigned to these three roles. The enabling intervals of all the roles involved in the purchase order workflow process are shown in Figure 3(b).

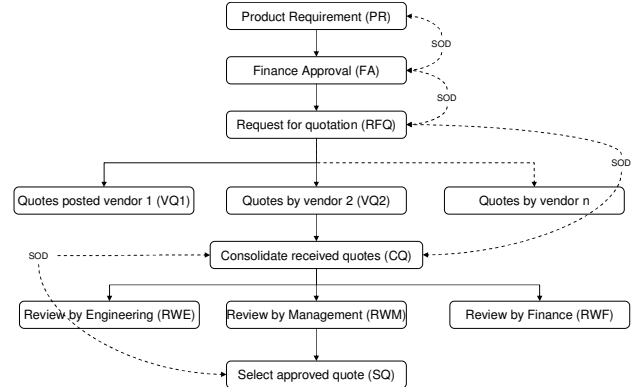


Figure 2. Purchase order processing work flow

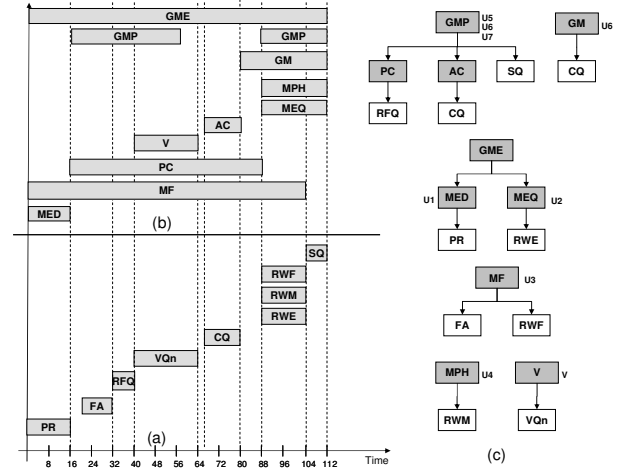


Figure 3 (a). Task duration and inter-task dependency constraints of the purchase order workflow of figure 2. (b) Role enabling intervals (c) Role hierarchy, user-role and task-role assignments specified in the organization's access control policy

Suppose in the workflow instance of Figure 2, users u_5 , u_6 and u_7 are scheduled to perform tasks RFQ, CQ, and SQ respectively. After execution of the task RFQ by user u_5 , assume that u_6 is unable to activate the role GMP and its junior roles. However, u_6 can activate the role GM during its enabling interval. The role GM is not enabled during the scheduled execution time of the task CQ. Therefore, user u_6 cannot perform task CQ at its scheduled time. In this case, either the task CQ needs to be delayed until the role GM gets enabled, or CQ needs to be reassigned to another user. Note that because of the SoD constraint between tasks CQ and RFQ, u_5 who has executed task RFQ, cannot be assigned to task CQ. Similarly, user u_7 scheduled for task SQ cannot execute task CQ because of the SoD constraint between CQ and SQ. Another option for adaptation is to relax one of the SoD constraints. In order to find the best solution, all these options need to be evaluated based on the desired optimality criterion.

Temporal constraints for role enabling derived from Rule 1.
1: $t_s^{MED}=0$; **2:** $t_f^{MED}=16$; **3:** $t_s^{MF}=8$; **4:** $t_f^{MF}=130$; **5:** $t_s^{PC}=32$; **6:** $t_f^{PC}=56$; **7:** $t_s^V=34$; **8:** $t_f^V=64$; **9:** $t_s^{AC}=60$;
10: $t_f^{AC}=90$; **12:** $t_s^{MEQ}=50$; **13:** $t_f^{MEQ}=100$; **14:** $t_s^{MPH}=50$; **15:** $t_f^{MPH}=150$; **16:** $t_s^{GMP}=0$; **17:** $t_f^{GMP}=200$; **18:** $t_s^{GM}=63$; **19:** $t_f^{GM}=130$; **20:** $t_s^{GME}=0$; **21:** $t_f^{GME}=112$;
Task duration constraints derived from Rule 2.
22: $t_f^{PR} - t_s^{PR} \square 10i^{PR}$; **23:** $t_f^{PR} - t_s^{PR} \square 8i^{PR}$; **24:** $t_f^{FA} - t_s^{FA} \square 12i^{FA}$; **25:** $t_f^{FA} - t_s^{FA} \square 4i^{PR}$; **26:** $t_f^{CQ} - t_s^{CQ} \square 14i^{CQ}$; **27:** $t_f^{CQ} - t_s^{CQ} \square 8i^{CQ}$; **27:** $t_f^{RFQ} - t_s^{RFQ} \square 8i^{RFQ}$; **28:** $t_f^{RFQ} - t_s^{RFQ} \square 4i^{RFQ}$; **29:** $t_f^{VQN} - t_s^{VQN} \square 32i^{VQN}$; **30:** $t_f^{VQN} - t_s^{VQN} \square 24i^{VQN}$;
Temporal constraints between role enabling and task execution for role MED and task PR derived from Rule 3.
31: $t_s^{MED} - t_s^{PR} \square 100000(1 - s_{MED}^{PR})$; **32:** $t_s^{PR} - t_s^{MED} \square 100000s_{MED}^{PR}$; **33:** $t_f^{PR} - t_f^{MED} \square 100000(1 - f_{MED}^{PR})$;
34: $t_f^{MED} - t_f^{PR} \square 100000f_{MED}^{PR}$; **35:** $\square_{MED}^{PR} = s_{MED}^{PR} f_{MED}^{PR}$
User-task authorization constraints for task PR derived from Rule 4.
36: $u_1^{PR} + u_2^{PR} - i^{PR} \square 0$; **37:** $u_1^{PR} - i^{PR} \square 0$; **38:** $u_2^{PR} - i^{PR} \square 0$;
User-role activation constraint for execution of task PR derived from Rule 5.
37: $\square_{GME}^{PR} u_1^{GME} + \square_{MED}^{PR} u_1^{MED} - u_1^{PR} \square 0$; **38:** $\square_{GME}^{PR} u_2^{GME} + \square_{MED}^{PR} u_2^{MED} - u_2^{PR} \square 0$;
Temporal dependency constraint c_1 between tasks RQE and FA derived from Rule 6''
39: $c_1(t_s^{FA} - t_f^{PR} - 5) \square 0$; **40:** $c_1(t_s^{FA} - t_f^{PR}) \square 0$;
Task-specific SoD c_2 between tasks SQ and CQ derived from Rule 7.
41: $c_2(u_5^{SQ} + u_5^{CQ} - 1) \square 0$; **42:** $c_2(u_6^{SQ} + u_6^{CQ} - 1) \square 0$; **43:** $c_2(u_7^{SQ} + u_7^{CQ} - 1) \square 0$;
44: $u_1^{PR}=1$; **45:** $u_3^{FA}=1$; **46:** $u_5^{RFQ}=1$

Figure 4. MIP translation of the purchase order processing work flow

The MIP based technique discussed in Section 4 can be used for determining optimal adaptation of the above workflow instance. The MIP constraints for this workflow instance are shown in Figure 4. In this figure, equations 1-21 specify the enabling time of different roles involved in the workflow process. The inequalities 22-30 define the task duration requirements for the workflow process. The temporal constraints between enabling of the role MED and the task PR is represented using inequalities 31-35. These inequalities specify that the enabling interval of role MED contains the execution interval of task PR ($\square_{MED}^{PR}=1$) if the following two conditions hold: i) MED is enabled prior to the execution of PR ($s_{MED}^{PR}=1$), and ii) MED remains enabled until PR finishes ($f_{MED}^{PR}=1$). The

inequalities 36-40, derived from Rules 4 and 5, specify that only u_1 and u_2 are authorized for task PR and these users can execute this task by assuming the role GME or the role MED. The temporal dependency constraints between the tasks PR and FA are captured by the inequalities 39 and 40, which specify that task PR precedes task FA, and FA must execute within five time units after completion of the task PR. The MIP constraint for task specific SoD between SQ and CQ is implied by the inequalities 41-43. Finally the status of the workflow till the execution of task RFQ by user u_5 is given by the equations 44-46.

Depending on the application requirements for adaptation, the MIP problem of Figure 4 can be solved with appropriate objective function. Table 5 shows the objective function and the resulting values of the MIP problem of Figure 4 with three different optimality measures. The first formulation in Table 5 considers minimization of the overall task execution delay without relaxing any workflow or policy constraints. With this optimality criterion, tasks CQ, RWF, RWM, RWE, and SQ need to be delayed by five time units from their scheduled times in the original workflow specification. In the second formulation, shown in Table 5, the optimality criterion is to minimize user-task reassignment with zero task scheduling delay and without relaxing any workflow and policy constraints. In this case, the tasks CQ and SQ cannot be executed by the assigned users u_6 and u_7 respectively. For successful completion of the workflow, task SQ is assigned to user u_5 and CQ is assigned to user u_7 in the reconfigured workflow instance. The third formulation considers minimization of both user-task assignment and overall task scheduling delay. In this case, the original user-task assignment is retained in the reconfigured workflow provided the overall scheduling delay is less than ten time units. This is reflected in the objective function by the weights assigned to the user-task and delay variables. The weight assigned to each delay variable is one-tenth of the weight assigned to the user-task variable, as shown in Table 5. The solution in this case is the same as found in the second formulation discussed above. However, allowing larger delays may change the user assignment. For instance, allowing an aggregate delay of thirty time units retains the original user-task assignment with an overall delay of 25 time units.

Table 5. Objective functions of purchase order processing workflow with different optimality criteria

1	Optimality Criterion	Minimize overall task execution delay without relaxing any workflow or policy constraints.
	Objective Function	Maximize $-d^{PR} - d^{FA} - d^{RFQ} - d^{VQN} - d^{CQ} - d^{RWF} - d^{RWM} - d^{RWE} - d^{SQ}$
	Additional Constraints	$d^{PR} = t_s^{PR}$; $d^{FA} = t_s^{FA} - 18$; $d^{RFQ} = t_s^{RFQ} - 32$; $d^{VQN} = t_s^{VQN} - 40$; $d^{CQ} = t_s^{CQ} - 68$; $d^{RWF} = t_s^{RWF} - 88$; $d^{RWM} = t_s^{RWM} - 88$; $d^{RWE} = t_s^{RWE} - 88$; $d^{SQ} = t_s^{SQ} - 104$; All elements of the constraint vector c and task execution indicator variables are set to one.
	Objective Function Values	$d^{PR} = 0$; $d^{FA} = 0$; $d^{RFQ} = 0$; $d^{VQN} = 0$; $d^{CQ} = 5$; $d^{RWF} = 5$; $d^{RWM} = 5$; $d^{RWE} = 5$; $d^{SQ} = 5$; Aggregate delay = 25
2	Optimality Criterion	Minimize user to task reassignment with zero task scheduling delay and without relaxing any workflow and policy constraints.
	Objective Function	Maximize $u_1^{PR} + u_3^{FA} + u_5^{RFQ} + u_v^{VQN} + u_6^{CQ} + u_3^{RWF} + u_4^{RWM} + u_2^{RWE} + u_7^{SQ}$
	Additional Constraints	All delay variables are set to Zero All elements of the constraint vector c and task execution indicator variables are set to one.
	Objective Function Values	$u_1^{PR} = 1$; $u_3^{FA} = 1$; $u_5^{RFQ} = 1$; $u_v^{VQN} = 1$; $u_6^{CQ} = 0$; $u_3^{RWF} = 1$; $u_4^{RWM} = 1$; $u_2^{RWE} = 1$; $u_7^{SQ} = 0$; Tasks CQ and SQ not executed by the users assigned in the original specification.
3	Optimality Criterion	Minimize user to task reassignment with minimum delay and keeping the overall task scheduling delay within 10 time units and without relaxing any workflow and policy constraints.
	Objective Function	Maximize $u_1^{PR} + u_3^{FA} + u_5^{RFQ} + u_v^{VQN} + u_6^{CQ} + u_3^{RWF} + u_4^{RWM} + u_2^{RWE} + u_7^{SQ} - 0.1d^{PR} - 0.1d^{FA} - 0.1d^{RFQ} - 0.1d^{VQN} - 0.1d^{CQ} - 0.1d^{RWF} - 0.1d^{RWM} - 0.1d^{RWE} - 0.1d^{SQ}$
	Additional Constraints	$d^{PR} = t_s^{PR}$; $d^{FA} = t_s^{FA} - 18$; $d^{RFQ} = t_s^{RFQ} - 32$; $d^{VQN} = t_s^{VQN} - 40$; $d^{CQ} = t_s^{CQ} - 68$; $d^{RWF} = t_s^{RWF} - 88$; $d^{RWM} = t_s^{RWM} - 88$; $d^{RWE} = t_s^{RWE} - 88$; $d^{SQ} = t_s^{SQ} - 104$; All elements of the constraint vector c and task execution indicator variables are set to one.
	Objective Function Values	$u_1^{PR} = 1$; $u_3^{FA} = 1$; $u_5^{RFQ} = 1$; $u_v^{VQN} = 1$; $u_6^{CQ} = 0$; $u_3^{RWF} = 1$; $u_4^{RWM} = 1$; $u_2^{RWE} = 1$; $u_7^{SQ} = 0$; Tasks CQ and SQ not executed by the users assigned in the original specification. All delay variables are ZERO

6. Related Work

Several research proposals have been made for secure workflow composition and management [4], [6], [14], [1], [2], [13]. A major focus of these research efforts is either on generating languages and formalism for incorporating security and authorization constraints in workflow applications or on verifying the consistency and safety of workflow specifications. The problem of managing context dependent and adaptive workflows has not been adequately addressed in literature. Bertino et. al. [4] have proposed a framework for specification and enforcement of authorization constraints in workflow management systems. The workflow authorization constraints in this framework are specified using temporal role-based access control (TRBAC) model. An important component of this framework is the workflow planner that assigns users to various tasks. The planner is invoked before the workflow execution begins to generate the initial assignment for workflow

execution. However, the initial plan can be changed during runtime to account for exceptions. In this case the planner is re-invoked to select a new plan from a feasible set of alternate plans. While selecting an alternate plan, the plan which assigns blocked tasks to users assuming junior roles is given priority over those plans that assign the blocked tasks to users assuming senior roles. No other optimality measures such as minimizing scheduling delays or constraint relaxation is considered in [4]. Crampton [6] has proposed a reference monitor system for management of constrained workflows. The reference monitor is used to verify the constraint satisfiability and completion requirement of a workflow instance with the given set of authorized users. Accordingly, various user-task assignments satisfying the given workflow authorization constraints can be determined. However, the system does not consider any criterion for assigning tasks to users when multiple assignments are possible.

7. Conclusion

In this paper we proposed a framework for dynamic adaptation of time-dependent workflows based on the authorization and security constraints, execution status, and environmental context. The proposed framework uses a mixed integer programming (MIP) based technique for finding the best possible workflow adaptation according to the pre-defined optimality criterion. The proposed technique is generic and can be tuned to a variety of optimality measures including minimization of task execution delays, minimization of user-task reassignment, and minimization of constraint relaxation.

The high computational overhead of the MIP approach for optimal solution is a major concern in workflow applications that have real-time scheduling constraints. Various approximation algorithms such as Lagrangian relaxation, tabu search, and simulated annealing can be used to solve the underlying MIP problem for near optimal solution in polynomial time. Studying the performance trade-off between these heuristics is an interesting problem that needs further research considerations.

8. References

- [1] W-K. Huang and V. Atluri, "SecureFlow: A Secure Web-enabled Workflow Management System," *4th ACM Workshop on Role-based Access Control*, October, 1999.
- [2] V. Atluri and W-K. Huang, "A Petri Net Based Safety Analysis of Workflow Authorization Models," *Journal of Computer Security, Volume 8, Issue 2/3*, 2000.
- [3] R. D. Holowczak, S. A. Chun, F. J. Artigas, V. Atluri, "Applications: Customized Geospatial Workflows for E-Government Service," in *Proc. Of 9th ACM International Symposium on Advances in Geographic Information Systems*, 2001.
- [4] E. Bertino, E. Ferrari, and V. Atluri, "The Specification and Enforcement of Authorization Constraints in Workflow Management Systems," *ACM Transactions on Information and System Security, Vol. 2, No. 1, 65-104*, 1999.
- [5] E. Bertino, P. A. Bonatti, E. Ferrari, "TRBAC: A Temporal Role-based Access Control Model," *ACM Transactions on Information and System Security, 4(3):191-233*, August 2001.
- [6] J. Crampton, "A Reference Monitor for Workflow Systems with Constrained Task Execution", in *10th ACM Symposium on Access Control Models and Technologies SACMAT 2005*.
- [7] E. Chang, E. Guatama, and T.S. Dillon, "Extended Activity Diagrams for Adaptive Workflow Modelling," in *Proc. ISORC*, 2001.
- [8] K. A. Delic. L. Douillet, and U. Dayal, "Towards an Architecture for Real-Time Decision Support Systems: Challenges and Solutions," in *Proc. International Symposium on Database Engineering and Applications*, 2001.
- [9] H. A. James, K. A. Hawick, and P. D. Coddington, "An Environment for Workflow Applications on Wide-Area Distributed Systems," in *Proc. International Conference on System Sciences*, 2001.
- [10] J. B. D. Joshi, E. Bertino, U. Latif, and A. Ghafoor, "Generalized Temporal Role Based Access Control Model," *IEEE Transaction on Knowledge and Data Engineering*, Vol. 17, No. 1, January 2005, pages. 4-23.
- [11] B. Shafiq, A. Samuel, H. Ghafoor, "A GTRBAC Based System for Dynamic Workflow Composition and Management", in proceedings of the *Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2005)*.
- [12] S. K. Shrivastava, S. M. Wheeler, "A Transactional Workflow based Distributed Application Composition and execution Environment," in *Proc. Eighth ACM SIGOPS European Workshop on Support for Composing Distributed Applications*, 1998.
- [13] T. Ahmed and A. R. Tripathi, "Static Verification of Security Requirements in Role Based CSCW Systems," *8th ACM Symposium on Access Control Models and Technologies (SACMAT 2003)*, 196--203, June, 2003.
- [14] S. S. Yau, H. Davulcu, S. Mukhopadhyay, D. Huang and Y. Yao, "Adaptable Situation-Aware Secure Service-Based (AS³) Systems" in proceedings of the *Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2005)*.
- [15] R. Sandhu, E. J. Coyne, H. L. Feinstein, C. E. Youman, "Role Based Access Control Models", *IEEE Computer*, Vol. 29, No 2, February 1996.
- [16] R. M. Sivasankaran, J. A. Stankovic, D. Towsley, B. Purimetla and K. Ramamritham. Priority Assignment in Real-time Active Databases. *The VLDB Journal*, 5, pp. 19-34, 1996.
- [17] J. Joshi. A Generalized Temporal Role Based Access Model For Developing Secure Systems. *PhD Thesis*. Purdue University, West Lafayette, IN. 2003.
- [18] J. F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11), pp.832-843, 1983.