Design of an Integrated Role-Based Access Control Infrastructure for Adaptive Workflow Systems

Nanjangud C. Narendra

IBM Software Labs, Bangalore, India

With increasing numbers of organizations automating their business processes by using workflow systems, security aspects of workflow systems has become a heavily researched area. Also, most workflow processes nowadays need to be adaptive, i.e., constantly changing, to meet changing business conditions. However, little attention has been paid to integrating Security and Adaptive Workflow. In this paper, we investigate this important research topic, with emphasis on Role-Based Access Control (RBAC) in Adaptive Workflow. Based on our earlier work on a 3-tier adaptive workflow architecture, we present the design of a similar 3-tier RBAC infrastructure, and we show that it conceptually mirrors our adaptive workflow architecture. We also describe the mappings between them, and we show how this mapping can be used to manage organizational RBAC constraints when the workflows are being adapted continuously. We illustrate our ideas throughout the paper with a simple yet non-trivial example.

Keywords: security, role based access control, adaptive workflow.

1. Introduction

Increasing numbers of organizations are automating their business processes by installing and using workflow systems. This has raised several research issues regarding security aspects of these workflow systems. Hence Secure Workflow has become a fertile research area [Atluri, et.al., 2000; Bertino, et.al., 1999; Cholewka, et.al,. 2000; Nyanchama and Osborn, 1999). However, most workflow processes nowadays need to be adaptive, i.e. constantly changing, to meet changing business conditions. In this paper, we address this important topic, with emphasis on Role-Based Access Control (RBAC). Based on our earlier work on a 3-tier adaptive workflow architecture (Narendra, 2000), we define a similar 3-tier RBAC infrastructure that mirrors our adaptive workflow architecture. We also describe the mappings between them, and we show how this mapping can be used to manage organizational RBAC constraints when the workflows are being adapted continuously. In fact, we believe that ours is the first attempt at integrating Security and Adaptive Workflow.

This paper is organized as follows. In the next Section, we provide an introduction to Workflow and Adaptive Workflow, where we present our 3-tier adaptive workflow architecture. In Section 3, we present the latest research results in Secure Workflow on which our work is based. The main result of this paper, i.e., our 3-tier RBAC infrastructure, is presented in Section 4. Section 5 concludes the paper with directions for future research. We also illustrate our ideas throughout the paper using, a simple yet non-trivial example.

2. Workflow Preliminaries

2.1. Introduction to Workflow

Our basic workflow model is based on the graph-theoretic notions in the OpenPM (Shan, et.al., 1997) process model. We assume that each workflow instance is a directed graph W = <N,E>, where N is the set of nodes and E is the set of edges. Each edge *e* in E is a tuple of the form $< n_{begin}, n_{end} >$, where the edge is directed from n_{begin} to n_{end}.

We first define two unique nodes — START and END nodes; for each workflow, there will be only one of each kind. A START node has no predecessor, and an END node has no successor.

The workflow graph is assumed to obey two simple conditions:

- Every node in the graph is reachable from the START node, i.e., there is a path from the START node to every other node in the graph
- The END node is reachable from every node in the graph, i.e., there is a path from the node to the END node

Nodes are of two kinds — *work nodes* and *route nodes*. Work nodes are nodes that actually perform the activities in the workflow. Route nodes are nodes whose only function is to evaluate rules (i.e., boolean conditions) and, depending on the result of the evaluation, these nodes direct the flow of control to specific work nodes or route nodes.

For each work node, we assume the existence of specific data structures that capture the data flow information with the node. In other words, each work node is assumed to perform certain operations on data and artifacts (e.g., complete a form, collate and summarize data, etc.). These operations, together with the data flow, define the flow of control throughout the workflow; and this is a consequence of the workflow design.

In our workflow model, edges are of four kinds — *forward edge, loop edge, soft-sync edge and strict-sync edge*. Forward edges depict the normal workflow execution, which is in a forward direction. Loop edges are backward pointing edges that are used to depict the repeated execution of loops.

The "sync" edges are quite different from forward edges, in that these edges are used to

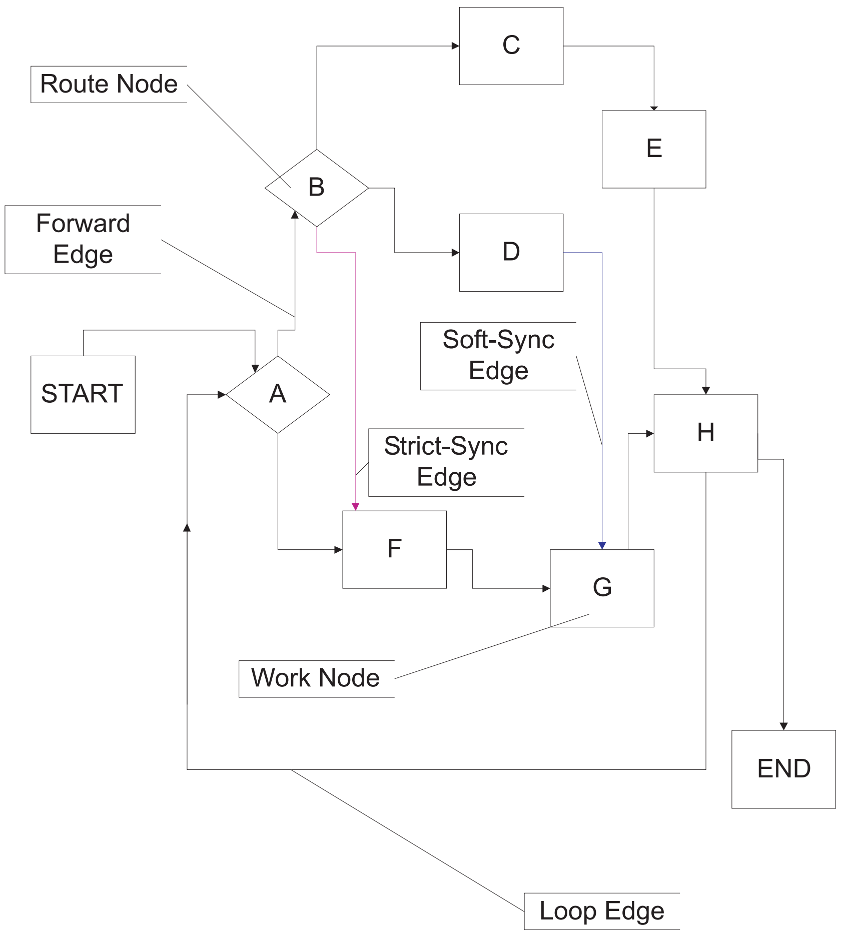

Fig. 1. Workflow Graph.

support synchronizations of tasks from *different* parallel branches of a loop. There are two types:

- A "soft-sync" edge from n_1 to n_2 is used to signify a "delay dependency" between n_1 and n_2, i.e., n_2 can only be executed if n_1 is either completed or cannot be triggered any more. This type of synchronization does not require successful completion of n_1.

- A "strict-sync" edge from n_1 to n_2 requires that n_1 successfully complete before n_2 executes.

Clearly, the use of such edges must satisfy some conditions:

- Redundant control flow dependencies between nodes and loops should be avoided

- A sync edge may not connect a node from inside a loop body with a node not contained within that loop

An example of workflow graph that illustrates all these definitions, is shown in Figure 1. The arcs depict the flow of the data and artifacts from the START to the END node.

We can also define states for each node, depending on where it is during the workflow execution. Hence we define the following states for nodes: NOT-ACTIVATED, ACTIVATED, DONE, FAILED, SUSPENDED.

The definition of these states naturally leads to the conclusion that we can define the state transition diagram that each node has to obey. Indeed, this means that the state of every node in the workflow instance is governed by the transition rules in the state transition diagram. This diagram is given in Figure 2.

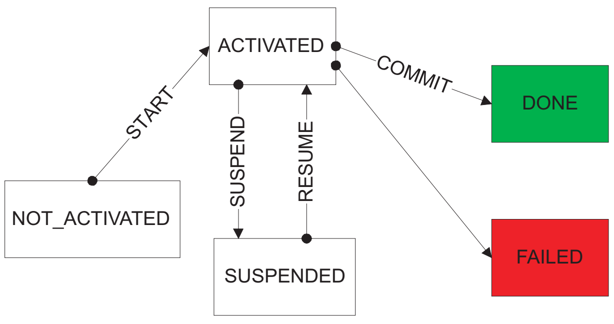

Fig. 2. State Transition Diagram.

2.2. Adaptive Workflow

In (Narendra, 2000), we developed a 3-tier approach to adaptive workflow management. Essentially, we recognized that a workflow needs to have a schema, i.e., a basic definition of its structure before it can be instantiated and executed. Moreover, since workflow is typically executed in business organizations, it is essential that it meet certain business goals. Therefore we see that business goals (essentially arising out of **planning**) lead to workflow **schema** being defined, out of which a workflow **instance** is generated for execution.

Hence we see that workflow adaptivity is basically of three types:

- **Adaptivity at instance level**: here, only the workflow instances need to be modified, perhaps to make them more efficient, or to make them easier to execute, etc. The modules that implement this are:
  - *Basic Workflow Model* — stores the instances of our workflow model
  - *Workflow Change Model* — stores the constructs for specifying dynamic workflow changes
  - *Workflow Change Verifier* — performs the syntactic and semantic checking for workflow adaptation
  - *Workflow–Schema Interface Manager* — interface module that interacts with the Schema Layer

- **Adaptivity at schema level**: here, the workflow schema will need to be modified – this would arise if certain "ways of doing things" change, and will necessitate radical modifications to all the workflow instances of the schema in question. The modules that implement this are:
  - *Schema Handler* — creates, versions and stores the workflow schema
  - *Schema Migration Manager* — handles migration of workflow instances between schemas, and interfaces with the Schema Handler
  - *Schema-Planning Interface Manager* — interface module that interacts with the Planning Layer

- **Adaptivity at planning/goal level**: here, the goals may themselves have to be modified in response to changing environmental conditions – this could cause radical changes in the schema themselves, which could cause disruptive changes in the workflow instances. The modules that implement this are:

- *Organizational Workflow Repository* — organization-wide repository of all workflow processes stored into the system; stores the workflow schemas and their instances, and also stores the goals that led to the creation of the workflow in the first place.
- *Goal Specification Modul*e — through this module, the workflow designer first enters the goals that the workflow has to satisfy. He/she can then look into the *Organizational Workflow Repository*, for reusing any workflows from the past that can satisfy — either fully or partially — the goals that he/she has specified.
- *Workflow Design Module* — with this module, and with the goal and workflow information from the *Goal Specification Module* and the *Organizational Workflow Repository*, the workflow designer can design the workflow schema. He/she will then check it into the *Organizational Workflow Repository*. The schema will be transferred into the *Schema Handler* via the *Schema-Plannng Interface Manager* where it will be appropriately versioned and stored. The appropriate version number is then communicated to the user.

The architecture is pictorially depicted in Figure 3.

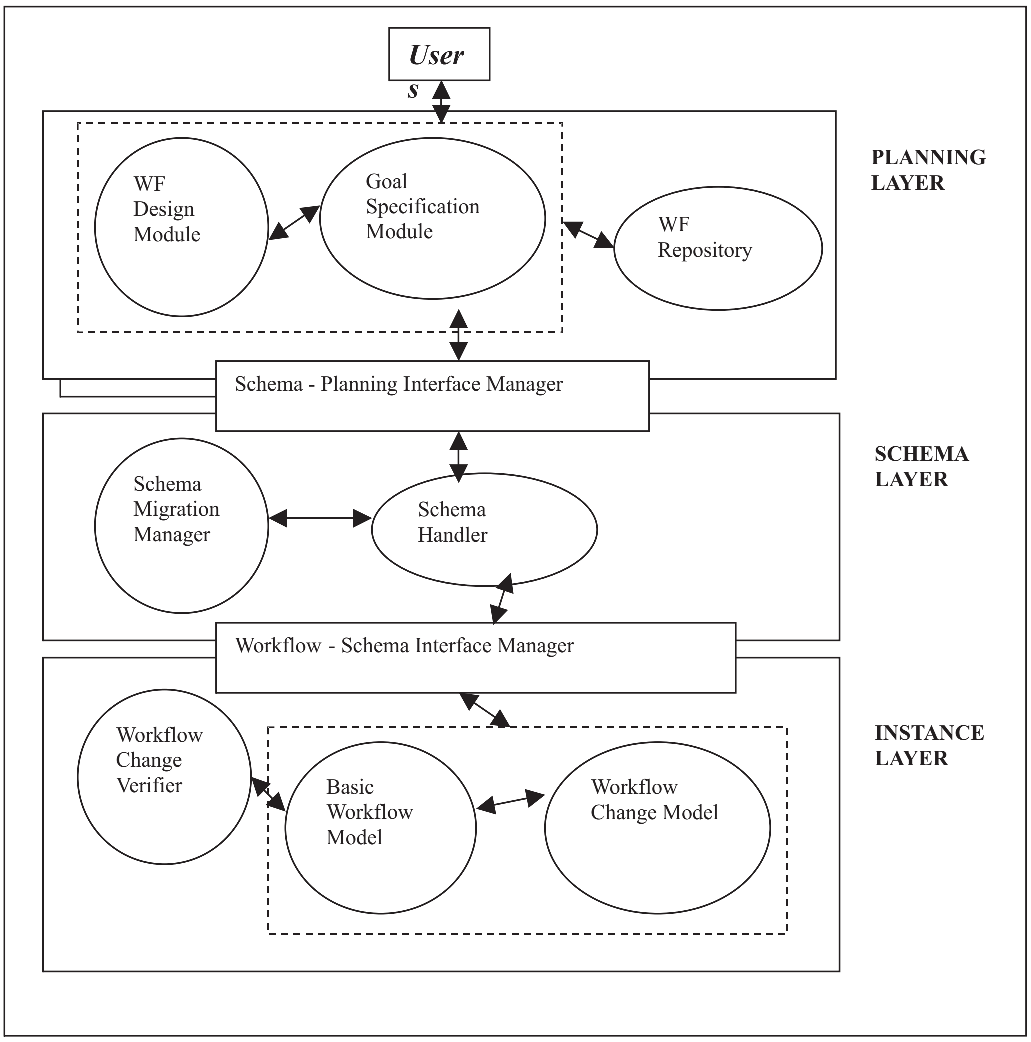

Fig. 3. Three-Tier Adaptive Workflow Architecture.

3. Role-Based Access Control (RBAC) in Workflow

3.1. Preliminaries

The most well-known approach towards security administration has been the Role-Based Access Control (RBAC) model (Sandhu and Samarati, 1994). This model allocates access rights to users based on the Roles that they perform in the organization; these access rights are also called "privileges". Roles can also be organized in a hierarchy — typically a partial order. In this hierarchy, Roles at higher levels are assumed to "inherit" the access rights of Roles at lower levels in the hierarchy, in addition to the access rights that they already possess.

In addition to this hierarchy, constraints can also be defined on how these access rights are granted to particular Roles. Some of the most common constraints are Separation of Duty constraints, of which there are two types:

- *Static Separation of Duty (SSOD)* — these constraints impose static restrictions such as "the role R cannot have access rights A1 and A2 simultaneously"
- *Dynamic Separation of Duty (DSOD)* — these constraints impose dynamic restrictions such as "if role R1 has executed task T1, then R1 cannot execute task T2"

In other words, SSOD constraints can be evaluated without executing the workflow, whereas DSOD constraints evaluation can only be performed at run time.

Hence in (Sandhu and Samarati, 1994), the RBAC model has itself been modeled as a partial order lattice depicted in Figure 4:

- ○ RBAC0 — the basic RBAC model
- ○ RBAC1 — RBAC0 with role hierarchies
- ○ RBAC2 — RBAC0 with constraints
- ○ RBAC3 — combination of all of the above

An RBAC3-compliant version of RBAC has been presented in (Nyanchama and Osborn, 1999), which also presents a 3-tier RBAC model, consisting of User-Group assignments, User-Role assignments and Role-Privilege assignments. Unique features of this model are the following:

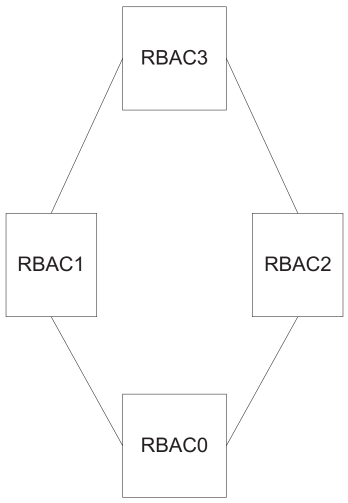

Fig. 4. RBAC Lattice.

- The notion of "direct" privileges (privileges that are owned solely by the Role) and "inherited" privileges (privileges owned by Roles that are junior to the Role in question)
- Algorithms for Role management, i.e., Role addition & Role deletion, and addition/deletion /modification of privileges associated with a Role
- The ideas of the conflicts that may arise in this model and how they are handled:
  - ○ *Role-role conflicts* — assigning two conflicting Roles to a user
  - ○ *Privilege-privilege conflicts* — assigning two conflicting privileges to a Role
  - ○ *User-role assignment conflicts* — assigning a Role to a user who is not allowed to play that role
  - ○ *Role-privilege assignment conflicts* — assigning a privilege to the Role which is not allowed to access that privilege

A sample Role Graph is presented in Figure 5. For the sake of completeness, every Role Graph has two optional "dummy" roles, MaxRole and MinRole, that are the senior and junior, respectively, of all Roles. Numbers next to each role are the direct privileges of that Role. For example: for VP1, the direct privileges are $\{9,10\}$, whereas the inherited privileges are $\{1,2,3,4,5,6\}$, which consist of the direct privileges of all Roles reporting directly or indirectly to VP1.

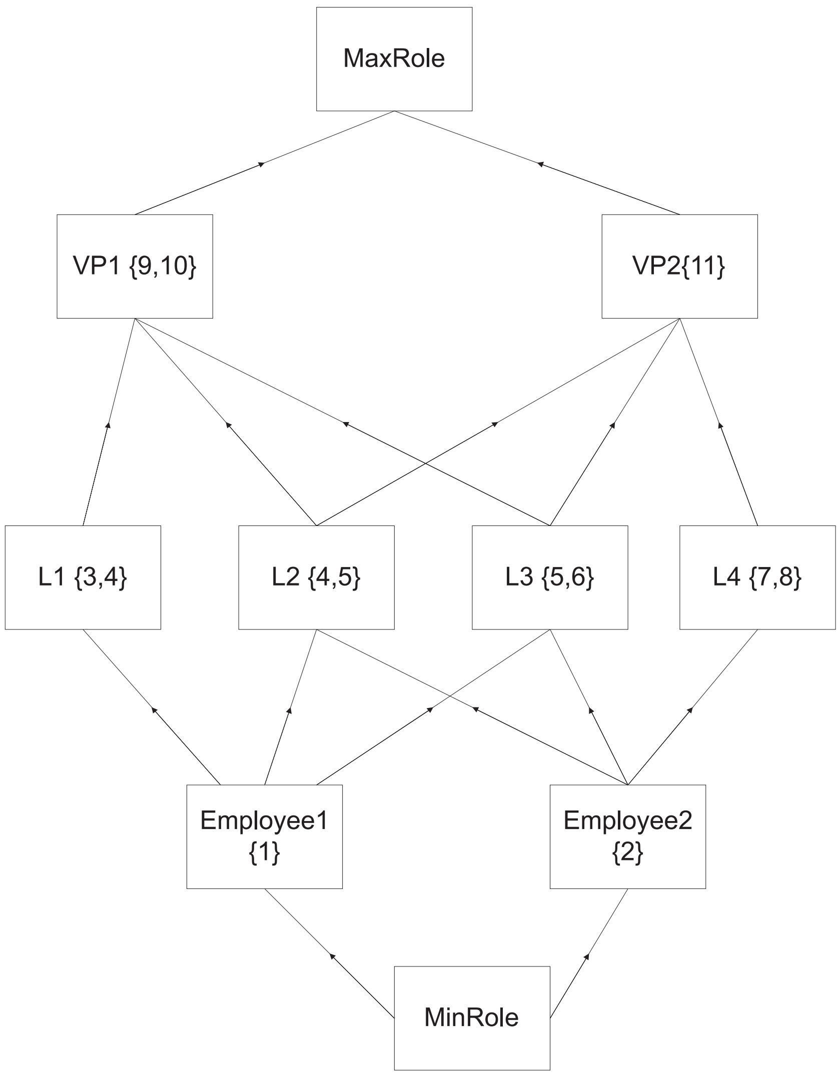

Fig. 5. Role Graph.

The following algorithms for Role management have been provided in (Nyanchama and Osborn, 1999):

- *Role Addition with only direct privileges* — a Role is added to the Role Graph along with its direct privileges only
- *Role Addition with inherited privileges* — a Role is added to the Role Graph along with its direct and inherited privileges
- *Role Deletion* — a Role is deleted from the Role Graph; this would also involve either deletion or re-allocation of the direct privileges associated with the Role
- *Privilege Addition* — a privilege is added to a Role
- *Privilege Deletion* — a privilege is deleted from a Role
- *Edge Insertion* — a new reporting relationship between a Role and one of its Senior Roles is added
- *Edge Deletion* — a reporting relationship between a Role and one of its Senior Roles is deleted.

All Addition algorithms also contain steps for checking for the 4 types of conflicts described above, and will abort with an error message in case a conflict is detected. The algorithms also incorporate mechanisms for detecting cycles in the Role Graph, in which case they will abort with an error message. For details, please see the original paper (Nyanchama and Osborn, 1999).

Please note that "negative authorizations" (for example, as specified in Damianou, et.al., 2001), viz., that a Role cannot access a particular privilege, are implicitly incorporated into the Role Graph by simply ensuring that the privilege in question does not appear along with the Role in the Role Graph. In other words, by default, a Role has only negative authorizations, unless and until any particular privileges are assigned to it.

The real-life applicability of the RBAC model has been validated in a real-life case study of a large European bank, which is presented in (Schaad, et.al., 2001).

In (Thomas and Sandhu, 1997), an RBAC-based model for workflow called Task-Based Authorization Control (TBAC) is presented. This model specifies, in a manner similar to that of RBAC, the tasks that each role is authorized to perform. A lattice structure similar to that of RBAC, i.e., TBAC0 through TBAC3 is also presented. This is taken further in (Cholewka, et.al., 2000), where a mapping between RBAC and a similar TBAC-like model are presented. In essence, TBAC works by setting users' access rights either at workflow definition time, or at workflow execution time before the workflow task is executed.

A similar model is presented in (Huand and Atluri, 1999). In this model, a Workflow Authorization Template is attached to each task in the workflow. This template grants appropriate authorizations only when the task starts and revokes it once the tasks has finished. More than one Template can be assigned to a task in case more than one user needs to execute the task.

The area of Multilevel Secure (MLS) Workflows is an offshoot of Secure Workflow, and is concerned with modeling workflows when different tasks in the workflow are at different security levels. More details can be found from [Atluri, et.al., 2000; Kang, et.al., 1999); however, MLS workflows are out of the scope of this paper.

In this paper, we assume the existence of a single Workflow Administrator who has the ability and the authority to define/adapt and deploy the workflows on the workflow system. This would, of course, raise the issue of having a hierarchy of workflow administrators, each with privileges over certain workflows (Sandhu and Munawer, 1997); however, this issue is out of the scope of this paper.

A very good overall introduction to security issues in workflow can be obtained from (Atluri, 2002).

Recently, the RBAC model has been extended to include dynamic aspects, i.e., Temporal RBAC (Bertino and Bonatti, 2001). This model supports periodic enabling and disabling of roles, possibly with individual exceptions for particular users.

3.2. Authorization Model for Secure Workflows

A comprehensive authorization model for supporting specification and enforcement of authorization constraints in workflows has been presented in (Bertino, et.al., 1999). Briefly, the model specifies a constraint specification language through which the workflow designer can specify, which roles and users are obliged or denied access to the workflow tasks, in the form of static (specified at definition time) and dynamic (specified at run time) constraints. These constraints can also include the DSOD and SSOD constraints as defined in Section 3.1.

This model provides for both positive and negative authorizations for roles (resp. users), which are referred to as "Obliged Roles (resp. Users)" and "Denied Roles (resp. Users)". The constraint specification results in the creation of a Constraint Base, whose consistency is checked for ensuring that the roles (and users) belonging to the "Obliged" category to execute a task, if any, do not belong to the set of roles (and users) belonging to the "Denied" category for that task, and vice versa.

Hence the model proposes an algorithm for user-task allocation, whose inputs are the following:

- The Global SSOD & DSOD Security Constraints are represented in the Constraint Base (CB)
- The User-Role Mapping
- The Role Graph as described in Section 3.1, which depicts the Role-Task Assignment
- The output of the algorithm is the user-task allocation for the workflow, which will be the

input to our algorithm to be described in Section 4.4.

For our purposes, we redefine the "Obliged Roles (resp. Users)" as "Authorized Roles (resp. Users)", where Authorized essentially refers to the direct privileges for tasks the Role (resp. user) can access. Hence we redefine "Denied Roles (resp. Users)" as those that do not belong to the "Authorized Roles (resp. Users)" set.

In this section, we have only briefly introduced the algorithm; for more details, please refer to the original paper (Bertino, et.al., 1999). We have also not described the SSOD and DSOD constraints more formally here, since they are already described in (Bertino, et.al., 1999); moreover, our emphasis is on the 3-tier RBAC infrastructure itself.

4. Our Approach to RBAC in Adaptive Workflow

Major motivation behind our approach is to try to answer the following question: given our 3-tier adaptive workflow architecture, what is the appropriate RBAC infrastructure that would "mirror" our adaptive workflow architecture? In other words, for each level of adaptivity-instance, schema, planning — what would be the appropriate RBAC-related modifications to make?

To answer this question, we revisit the 3-tier model of (Nyanchama and Osborn, 1999) described in Section 3.1. We recast this model as the following 3-tier RBAC model:

- Global SSOD & DSOD Constraints
- User-Role Assignment
- Role-Task Assignment

This model maps naturally onto our 3-tier adaptive workflow architecture:

- When *business goals* are set for workflow processes, the task of planning and scheduling these processes should take into account the global SSOD & DSOD constraints while the workflow schema is being defined
- For *workflow schema definition*, the most crucial RBAC-related action that needs to be performed, is to assign the appropriate roles to the users in the organization. This assignment is also based on an understanding of the overall business goals of the organization, and also on the kinds of tasks that each role would need to execute.
- Hence, actual role-task assignment is done at the time of *workflow instance definition*, i.e., when instances are created out of the schemas. This is similar in approach to the TBAC model as described in (Thomas and Sandhu, 1997) and introduced in Section 3.1.

Hence, depending on the nature of the adaptation, the appropriate RBAC-related assignments may need to be modified in order to ensure that the modified workflow process can be executed with the users available:

- For *changes at instance level*, only role-task assignments may need to be modified, in order to ensure that all tasks in the modified workflow instance can be performed. For example, junior managers can be temporarily provided with permission to approve certain purchases if the change in the workflow instance results in the non-availability of a senior manager to perform the approval task.
- For *changes at schema level*, **one** of the two types of changes can be made:
  - ○ User-role assignments may need to be modified instead of mere role-task assignment changes. For example, a particular set of individuals may be assigned the senior manager role, so that they can (permanently) approve the purchases. This modification could also affect the role-task assignments, which may also need to be changed to reflect the changed user-role assignments.
  - ○ Global SSOD & DSOD constraints may need to be modified.

The reason why user-role assignments have been mapped to the workflow schema level is that user-role assignments are typically more radical and far-reaching than role-task assignments in most organizations. For example, promoting a junior manager to the senior manager role means that he/she automatically has access rights to all the tasks that the senior manager can perform in all future workflow instances deriving from the adapted workflow schema. On the other hand, temporarily providing him/her with some of the senior manager privileges for only a particular workflow instance, has much less impact on the organization.

The case is similar for SSOD & DSOD constraint modification. If either of the above two types of changes is made, then the impact on the organization will be far greater than merely changing role-task assignments.

- For *changes of business goals resulting in widespread workflow schema changes*, the global SSOD & DSOD constraints may also need to be modified, **in addition to** modifying the user-role assignments. That is, these two types of changes are far more disruptive than either one of them acting alone. This is because, as explained above, global constraints typically define the very basic constraints under which the organization functions; changing these constraints could lead to major changes in the organization structure itself.

One example of change in a global constraint is that approval for certain purchases below a certain amount may no longer be required. Alternatively, an additional SSOD constraint may be introduced that restricts a person from being a junior manager and also from being granted permission to approve purchases above a certain limit. Such changes in constraints could also affect the user-role assignments, which could, in turn, affect the role-task assignments.

Of course, our model may be an over-simplification in many cases, but it provides one way to effectively manage RBAC-related inconsistencies that could arise due to workflow changes.

The example to be introduced later in this Section will illustrate these concepts and show the efficacy of our model.

4.1. Modified Adaptive Workflow Architecture

Based on the above mapping, the Planning Layer of our 3-tier adaptive workflow architecture (depicted in Figure 3) is modified as shown in Figure 6.

Since global SSOD & DSOD constraints are essentially limitations on how the workflow will meet the user-specified goals, they are derived from the goals via the Goal Specification Module, and are specified in the Security Module. In this module, the Global Constraints

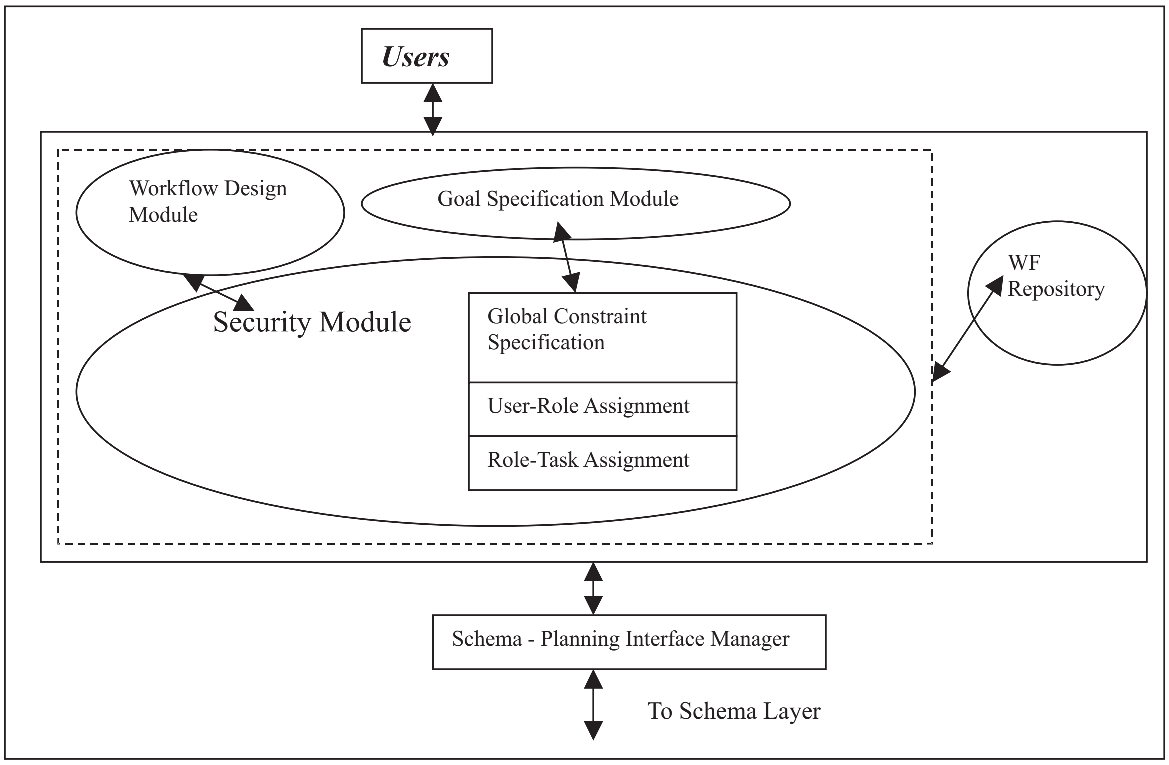

Fig. 6. Modified Planning Layer of Adaptive Workflow Architecture.

are specified in the Global Constraint Specification Module, which will also contain a consistency checker for ensuring consistency among the constraints. This will also involve setting the User-Role and Role-Task assignments.

4.2. Our Example

We select a simple yet non-trivial example of housing loan approvals at a bank (this is a simplified version of the process that the author himself had to go through while obtaining a housing loan at an Indian bank). We assume the existence of the following roles in the Housing Loans department: Department Manager, Junior Managers, and Clerks. The initially followed workflow for housing loan approval is shown in Figure 7.

Please note that in the original workflow, route nodes need to exist after work nodes #1, #3, #4 and #5, to allow for the possibility of the customer's application being found unsuitable at each of these nodes. For clarity, however, these have been elided from the diagram.

Please also note that the task #5 has been introduced since, as per regulations at most Indian banks, a portion of the house must already have been constructed before a housing loan can be granted. This is to ensure that the loan amount will be used only for the intended purpose, i.e., house building.

This workflow is then adapted in order to produce the following workflow as depicted in Figure 8.

Hence we see that the main adaptations are:

A Junior Manager's initial approval is additionally required, before additional processing can be done on the loan application.

The document examination and assessor's evaluation tasks are parallelized for efficiency; there is also a strict-sync edge from task #5 to task #4, signifying that House Assessment should be completed successfully before Documents Examination is executed.

The Role Graph for the Housing Loans Department is depicted in Figure 9 (the numbers next to each role are the tasks of the workflows of Figures 7 and 8 the role is allowed to execute, they are the privileges for the role).

One DSOD constraint in the Department is that the Clerk who has processed the customer's application, (i.e,. tasks #1, #2 and #3) cannot approve the documents or perform house assessment for the same customer (i.e., tasks #3a, #4 or #5). This constraint restricts the choice of who will perform tasks #3a, #4 and #5 in the event of a workflow adaptation if no Junior Manager or Department Manager is available.

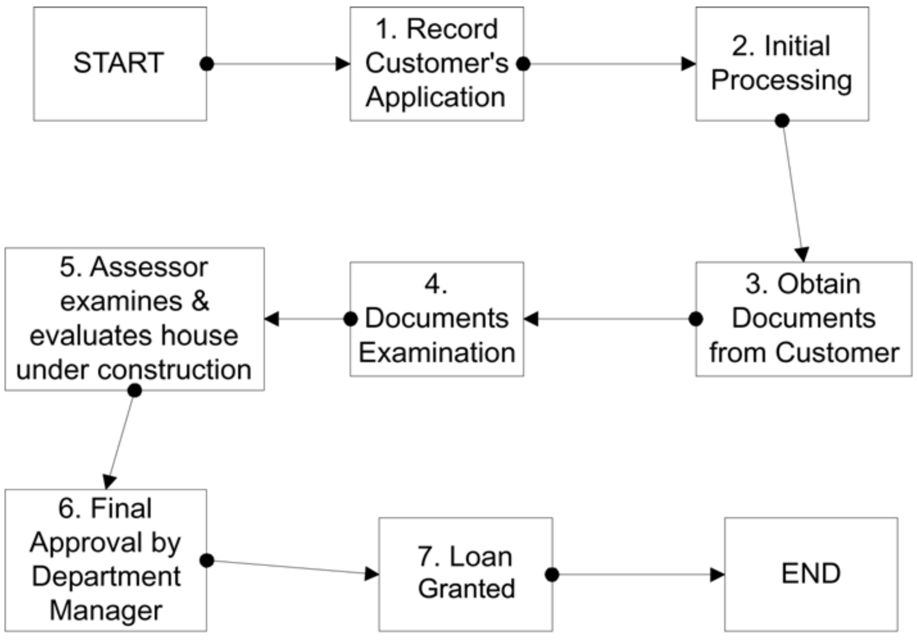
START
1. Record Customer's Application
2. Initial Processing
3. Obtain Documents from Customer
4. Documents Examination
5. Assessor examines & evaluates house under construction
6. Final Approval by Department Manager
7. Loan Granted
END

Fig. 7. Initially Followed Housing Loan Approval Workflow.

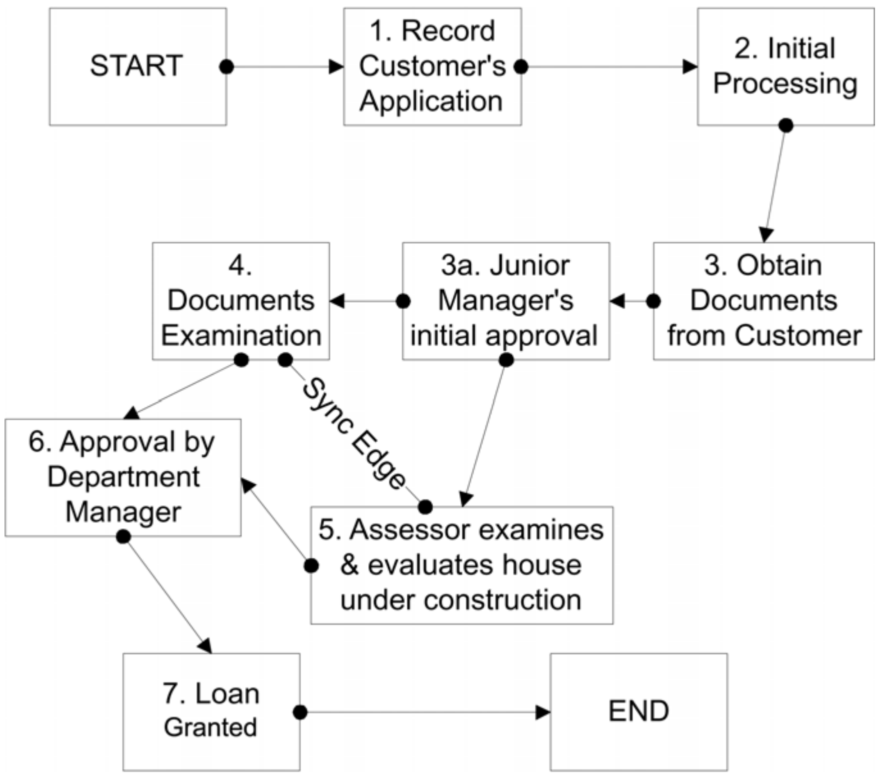
START
1. Record Customer's Application
2. Initial Processing
3. Obtain Documents from Customer
3a. Junior Manager's initial approval
4. Documents Examination
Sync Edge
5. Assessor examines & evaluates house under construction
6. Approval by Department Manager
7. Loan Granted
END

Fig. 8. Adapted Housing Loan Approval Workflow.

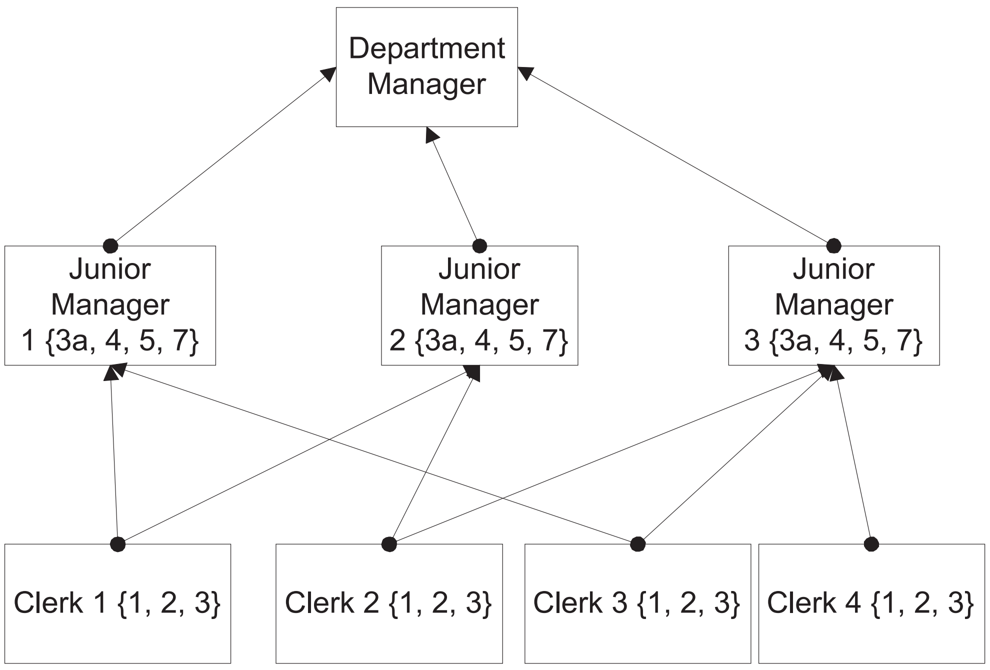
Department Manager
Junior Manager 1 {3a, 4, 5, 7}
Junior Manager 2 {3a, 4, 5, 7}
Junior Manager 3 {3a, 4, 5, 7}
Clerk 1 {1, 2, 3}
Clerk 2 {1, 2, 3}
Clerk 3 {1, 2, 3}
Clerk 4 {1, 2, 3}

Fig. 9. Role Graph for Housing Loans Department.

4.3. Modification Regions for Workflow Changes

A typical workflow change is a combination of two basic actions:

- Task deletion
- Task addition

When a task is deleted, if the tasks following this task are not changed in any way, then there is no impact on the rest of the workflow. However, if the succeeding tasks in the workflow are "brought forward" in order to take advantage of the task deletion, then all these tasks are potentially affected by this workflow change.

Tasks can be added in one of the following two ways:

- *In parallel* — this would mostly affect other tasks only if the user who is supposed to work on this new task is also needed for other tasks in the rest of the workflow.

- *Sequentially* — all succeeding tasks are potentially affected by this new task.

In any case, determining the tasks affected by the workflow change, which we call *Modification Region (MR)*, is a heuristic procedure that needs to be implemented by the user based on the semantics of the workflow. Moreover, as already mentioned in Section 3.1, we have already assumed that the Workflow Administrator has the ability to understand the workflow semantics so that he/she can identify the Modification Region for each workflow adaptation.

The MR is therefore the only part of the workflow that needs to be rescheduled due to workflow adaptation. In order to determine the MR, the Workflow Administrator can take advantage of the following features of our workflow model (see Section 2.1):

- *Sync edges* — tasks that are connected (even transitively) by (soft or strict) sync edges to the task being added or deleted, need to belong to the MR. Hence all other workflow tasks that are successors of the sync edge-connected task, also need to be evaluated as potential candidates to be included in the MR.

- *State Transitions* — all tasks that do not belong to either DONE or FAILED state are potential candidates to be included in the MR.

- *Loops* — all tasks belonging to the same loop as the task being added or deleted, also need to be considered as candidates for inclusion in the MR.

Hence, in our example, the tasks #4, #5, #6 and #7 form the MR, since they follow the added task #3a. Since tasks #1, #2 and #3 are already completed, they are in DONE state. Hence they do not belong to the MR.

4.4. Re-assigning Roles and Users to a Task

We assume the existence of the following:

- Role Graph as described in Section 3.1

- User-Role mapping, which maps users to the roles they perform on the Role Graph (according to the algorithm of Section 3.2)

As per the previous sub-section, when a task is either added or deleted, all tasks in the MR have to be rescheduled. Hence for each task in the MR, we need to do the following:

- Determine whether there are users possessing the roles assigned to the task (by invoking the algorithm of Section 3.2), who are available to execute the task. If so, the user is allocated to the task.

- If **not**, we adopt the following approach:

  ○ Let the role assigned to the task (as per the Role Graph) be R. We first search among the seniors of R, in increasing order of seniority in the Role Graph, as to whether any of them are available to execute this task (via inherited privileges). The search stops once a senior is found who is available to execute the task.

    ◇ If **not**, then we check among the juniors of each of the already checked seniors of the user in question. This continues recursively upwards in the Role Graph, starting at each of the seniors of the user in question, until the MaxRole role is reached. Of course, at any time, the search ends if an available user is found.

- If **not**, we search among the juniors of R, in decreasing order of seniority in the Role Graph. This is basically a "mirror image" of the search among the seniors, thus: at each junior, the seniors of the junior are also checked, and this also proceeds recursively down the Role Graph, until the MinRole role is reached. Again, the search stops once a junior is found who is available to execute the task.

In case the role of the available user cannot execute the task (as per the current Role-Task assignments), then the user needs to be temporarily allocated the privilege, so that he/she can execute the task. If this allocation becomes permanent, then the workflow change will be a schema-level change.

Please note that at each candidate role in both senior and junior searches, we also need to ensure that the Global SSOD & DSOD Constraints are obeyed. Otherwise, we move on to the next role and alert the workflow administrator as to the exact violations that would occur.

- If neither search produces a satisfactory result, the workflow administrator will have to take a decision manually. Alternatively, he/she can relax/remove some of the Global SSOD & DSOD Constraints so that some user can be found to execute the task.

In our example, the task #3a is added, while tasks #4 and #5 are parallelized, along with a strict-sync edge between them. In case a Junior Manager is not available to execute task #3a, then the above algorithm needs to be invoked in order to find out whether any other Junior Manager or the Department Manager is available for this task.

If not, then any available Clerk needs to be temporarily given the privilege of executing task #3a. In this case, if the clerk in question happens to be the one who processed the customer's application, then the DSOD constraint operating on this Clerk has to be removed to allow the Clerk to execute this task.

In case of a workflow schema change, this temporary privilege addition will become permanent. In other words, even Clerks will have the ability to perform initial approval, documents examination and house assessment for all future workflow instances and, in essence, they will become the equals in rank to Junior Manager. If, in addition, the constraint removal is also made permanent, then this will become a planning level change.

At first glance, such a re-assignment may seem to be rather unusual and may even appear detrimental to the security of an organization. However, temporary delegation of a task to a junior employee due to the absence of a senior employee is a common practice in most organizations.

The algorithm will then have to be implemented for the other tasks in the MR, viz., #4, #5, #6 and #7.

The overall algorithm is given in Fig. 10.

5. Conclusions and Future Work

In this paper, we have presented our approach to enforcing Role-Based Access Control (RBAC) in Adaptive Workflow. In line with our earlier work on a 3-tier adaptive workflow architecture, we have presented a 3-tier RBAC infrastructure for adaptive workflow, consisting of Global Constraints, User-Role Assignments and Role-Task Assignment Layers; this mirrors our adaptive workflow architecture, which consists of Planning, Schema and Instance layers, respectively. We have shown that temporary User and Role allocation changes are sufficient for instance level changes. Schema level changes demand that these allocation changes be made permanent, so that all future workflow instances deriving from the new workflow schema are affected by the newly changed allocations. In addition, Planning level changes require changes in the Global Constraints that define how the business operates itself.

We have also presented an incremental algorithm, leveraging off prior research in Role Based Access Control and Secure Workflow. With this algorithm, it is possible to implement re-allocations in the presence of workflow changes. The re-allocations need to be implemented only for those portions of the workflow (also known as Modification Regions) affected by the workflow change. Indeed, we believe that this is the first attempt at defining an RBAC model and infrastructure for adaptive workflow.

There are several opportunities for future work. Firstly, our RBAC model needs to be specified in more detail, so that it can be implemented and experimentally tested. In particular, safety and correctness algorithms (Mathews, 2001) need to be incorporated into the model. Moreover, a formal definition of our workflow language

```
**BEGIN**
Assume an initial allocation of users and roles to tasks in the workflow as per the algorithm
of Section 3.2. Also assume that a workflow change has been introduced.
Determine MR = {Affected Tasks} as in Section 4.3;
**for each** Task in MR **do**
{
    MR← MR\ {Task};/* The Task is removed from the set of Affected Tasks */
    IF the Task CAN be executed by the currently allocated user
    **then**
    {
        **return**; /* i.e., go to the next affected task in the MR */
}
IF NOT
    **then**
    {
        **S**: Implement Re-Assignment Algorithm from Section 4.4;
    **if** the Re-Assignment Algorithm is successful
    **then**
    {
        Assign the user and the role to the task as per the Re-Assignment Algorithm;
        **return**; /* i.e., go to the next affected task in the MR */
    }
    **if** NOT, **then**
    {
          Alert the Workflow Administrator, so that the Workflow Administrator can
          modify the SSOD & DSOD constraints in order to re-execute this algorithm;
          **go to** statement **S**;
          }
        }
    } **while** (there is another Task in the MR)
**END**
```

Fig. 10. Overall Algorithm.

and execution model, along with formal workflow goal specification will also be necessary for implementation purposes. Formal specification of the SSOD and DSOD constraints, along lines of (Bertino, et.al., 1999), also need to be incorporated into implementation. We should also incorporate more control-flow dependencies (such as strict-sync and soft-sync) into our workflow model, perhaps leveraging from the ACTA (Chrysanthis and Ramamritham, 1990) transaction model. A (semi-) automatic algorithm for MR determination can also be developed and implemented. Using ideas from (van der Aalst, et.al., 2001), we can also look at efficient and automatic MR determination algorithms specialized for specific workflow patterns.

Our technique can also be extended for workflows running in distributed and heterogeneous environments (Dellarocas and Klein, 1999; Herzberg, et.al., 2001; Kang, et.al., 2001; Narendra, 2001), managed by intelligent agents. Also, the European bank case study as reported in (Schaad, et.al., 2001), has brought one possible interesting extension to the basic RBAC model used in this paper, viz., defining groups of users for which roles can be assigned. The use of a general mechanism such as policy domains (Sloman and Twidle, 1994) needs investigation. Other extensions of the RBAC model, such as Team-Based Access Control (Georgiadis, et.al., 2001) and Set-Based Access Control (van den Akker, et.al., 2001), also need to be investigated. In particular, the applicability of TRBAC (Bertino and Bonatti, 2001) for Adaptive Workflow needs to be investigated.

6. Acknowledgments

The author wishes to thank his manager, Srinidhi Srinath and the SES Director, Padma Ravichanger, for their support.

References

[1] VAN DER AALST W.M.P., HOFSTEDE A.H.M., KIEPUSZEWSKI B., AND BARROS A.P., *"Workflow Patterns"*, 2001, available from `http://tmitwww.tm.tue.nl/staff/wvdaalst/Publications/p108.pdf`

[2] VAN DEN AKKER T., SNELL Q. O., AND CLEMENT M. J., "The YGuard Access Control Model: Set-Based Access Control", *ACM Workshop on Role Based Access Control, Proceedings of the Sixth ACM Symposium on Access Control Models and Technologies*, Chantilly, Virginia, USA, Pages 75–84, 2001, ISBN:1-58113-350-2.

[3] ATLURI V., "Security for Workflow Systems", *Information Security Technical Report*, Volume 6, Number 2, 2001, Elsevier Science, 2002, pp. 59–68; also available from `http://cimic.rutgers.edu/~atluri/workflow.pdf`

[4] ATLURI V., HUANG W-K., AND BERTINO E., "A Semantic-Based Execution Model for Multilevel Secure Workflows", *Journal of Computer Security*, Volume 8, Number 1, 2000, pp. 3–41; also available from `http://cimic.rutgers.edu/~atluri/jcs00a.ps`

[5] BERTINO E., AND BONATTI P.A., "TRBAC: A Temporal Role-Based Access Control Model", *ACM Transactions on Information and System Security*, Vol. 4, No. 3, August 2001, pp. 191–223.

[6] BERTINO E., FERRARI E., AND ATLURI V., "An Approach for the Specification and Enforcement of Authorization Constraints in Workflow Management Systems", *ACM Transactions on Information Systems Security*, February 1999, Vol. 1, No. 1; also available from `http://cimic.rutgers.edu/~atluri/tissec.ps`

[7] CHOLEWKA D. G., BOTHA R. A., AND ELOFF J. H. P., "A Context-Sensitive Access Control Model and Prototype Implementation", *Proceedings of IFIP/SEC 2000*, August 2000; also available from `http://www.petech.ac.za/secwflow/images/papers/SEC2000Cholewka.pdf`

[8] CHRYSANTHIS P. K., AND RAMAMRITHAM K., "ACTA: A Framework for Specifying and Reasoning about Transaction Structure and Behavior", in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 194–203, 1990.

[9] DAMIANOU N., DULAY N., LUPU E., SLOMAN M., "The Ponder Specification Language", *Workshop on Policies for Distributed Systems and Networks (Policy2001)*, HP Labs Bristol, 29–31 Jan 2001; also available from `http://www.doc.ic.ac.uk/~mss/Papers/Ponder-Policy01V5.pdf`

[10] DELLAROCAS C., AND KLEIN M., "Civil Agent Societies: Tools for inventing open agent-mediated electronic marketplaces", *Proceedings of the Workshop in Agent-Mediated Electronic Commerce (co-located with IJCAI'99)*, Stockholm, Sweden, July 1999; also available from `http://ccs.mit.edu/dell/civilagentsocieties.pdf`

[11] GEORGIADIS C. K., MAVRIDIS I., PANGALOS G., AND THOMAS R. K., "Flexible Team-Based Access Control Using Contexts", *ACM Workshop on Role Based Access Control, Proceedings of the Sixth ACM Symposium on Access Control Models and Technologies*, Chantilly, Virginia, USA, pp. 21–27, 2001, ISBN:1-58113-350-2

[12] HERZBERG A., MASS Y., MIHAELI J., NAOR D., AND RAVID Y., "Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers", 2001, available from `http://www.haifa.il.ibm.com/projects/software/e-Business/papers/Paper_Trust.pdf`

[13] HUANG W-K., AND ATLURI V., "SecureFlow: A Secure Web-enabled Workflow Management System", *4th ACM Workshop on Role-based Access Control*, October, 1999; also available from `http://cimic.rutgers.edu/~atluri/rbac99.ps`

[14] KANG M. H., FROSCHER J. N., EPPINGER B. J., AND MOSKOVITZ I. S., "A Multilevel Secure Workflow Management System", 1999, available from `http://www.itd.nrl.navy.mil/ITD/5540/publications/CHACS/1999/1999kang-IFIP99.pdf`

[15] KANG M. H., PARK J. S., AND FROSCHER J. N., "Access Control Mechanisms for Inter-Organizational Workflow", 2001, available from `http://citeseer.nj.nec.com/438991.html`

[16] MATTHEWS M. G., "Supporting Adaptive Change in B2B Coordination", *Proceedings of the International Conference on Artificial Intelligence*, Las Vegas, Nevada; 2001, also available from `http://mason.gmu.edu/~mmatthe1/Matthews_ICAI_2001.pdf`

[17] NARENDRA N. C., "Adaptive Workflow Management: An Integrated Approach and System Architecture", *Applied Computing 2000, Proceedings of the 2000 ACM Symposium on Applied Computing*, Villa Olmo, Via Cantoni 1, 22100 Como, Italy, March 19–21, 2000. ACM, 2000, ISBN 1-58113-239-5, Volume 2.

[18] NARENDRA N. C., "AdaptAgent: Integrated Architecture for Adaptive Workflow and Agents", it International Conference on Artificial Intelligence 2001, Special Session on Agent-oriented Software Architectures for B2B, Vol. II, (Ed.) H. R. Arabnia, CSREA press, June 25–28, 2001, Nevada, Las Vegas, USA, ISBN: 1-892512-79-3; also available from `http://lists.w3.org/Archives/Public/public-ws-chor/2003Apr/att-0275/adapt-agentB2B-ICAI.pdf`

[19] NYANCHAMA M., AND OSBORN S., "The Role Graph Model and Conflict of Interest", *ACM Transactions on Information and Systems Security*, Vol.2, No.1, February 1999, pp. 3–33; also available from `http://www.csd.uwo.ca/faculty/sylvia/conflict.ps`

[20] SANDHU R. S, AND MUNAWER Q., "The RRA97 Model for Role-Based Administration of Role Hierarchies", 1997, available from `http://citeseer.nj.nec.com/sandhu98rra.html`

[21] SANDHU R. S., AND SAMARATI P., "Access Control: Principles and Practice", *IEEE Communications*, Volume 32, Number 9, September 1994; also available from `http://citeseer.nj.nec.com/687.html`

[22] SCHAAD A., MOFFETT J., AND JACOB J., "The Role-Based Access Control System of a European Bank: A Case Study and Discussion", *ACM Workshop on Role Based Access Control*, 2001; 6th ACM Symposium on Access Control Models and Technologies (SACMAT 2001), May 3–4, 2001, Litton-TASC, Chantilly, Virginia, USA; also available from `http://citeseer.nj.nec.com/schaad01rolebased.html`

[23] SHAN M-C., DAVIS J., DU W., HUANG Y., "HP Workflow Research: Past, Present, and Future", *HP Labs Technical Report HPL-97-105*, August 1997; available from `http://www.hpl.hp.com/techreports/97/HPL-97-105.html`

[24] SLOMAN M. S., AND TWIDLE K. P., "Domains: A Framework for Structuring Management Policy", in *Network and Distributed Systems Management*, Addison-Wesley, 1994.

[25] THOMAS R., AND SANDHU R. S., "Task-based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-oriented Authorization Management", IFIP WG11.3, 1997; also available from `http://citeseer.nj.nec.com/thomas97taskbased.html`

Received: March, 2003
Revised: December, 2003
Accepted: December, 2003

Contact address:
Nanjangud C. Narendra
IBM Software Labs
Golden Enclave
Airport Road
560017 Bangalore
India
Phone: + 91 80 5094537
e-mail: `narendra@in.ibm.com`

DR. NANJANGUD C. NARENDRA is a Software Architect at IBM Software Labs India, in Bangalore, India. He is the author of over 30 papers and technical articles. His research interests are in the areas of information systems, workflow, agent technology, security and e-service management. He is a reviewer for IEEE Internet Computing, and has been a reviewer for ACM Symposium on Applied Computing and numerous other conferences on workflow and agents. The work for this paper was done while he was working as a Software Architect at Hewlett-Packard India Software Operations Ltd.