Secure workflow deployment in multi-organizational environments

Samiha Ayed, Nora Cuppens-Boulahia, Frédéric Cuppens

Secure workflow deployment in multi-organizational environments

Samiha Ayed (samiha.ayed@telecom-bretagne.eu)*
Nora Cuppens-Boulahia (nora.cuppens@telecom-bretagne.eu)*
Frederic Cuppens (frederic.cuppens@telecom-bretagne.eu)*

Abstract:
Workfow Management Systems (WFMS) are concerned with the control and coordination of operational business processes, called workflows. When workflow technology is deployed in domains where processes have simple coordinative requirements, the flow of control and data may be easily mapped onto process effectiveness. With the diversity of resources, subjects and activities in the system, ensuring a secure execution environment of the workflow becomes a critical issue. In this paper, we are interested in deploying a WFMS security policy. We investigate this issue either within intra or inter organizational workflows. The later case is more complex and requires more sophisticated control since it includes communications between different organizations.

Keywords: WFMS, OrBAC, Security Policy.

1 Introduction

A workflow is a process consisting of several tasks to be executed by enforcing specific constraints. WFMS are based on representing processes as workflows. The different tasks composing a workflow are interdependent and are communicating control information and data to each other. This means that access control must be synchronized with progression of the workflow execution. In addition, the execution of a task is related to the execution of precedent tasks. So an access control model is needed. On the other hand, workflow tasks can be executed within the same organization or within different ones. In the first case, the workflow security is controlled by the same organization which has the authority to manage the security of its different subjects and their interactions and to control the access to its different objects. In the case of different organizations, more complex management is necessary to ensure a secure execution of the workflow. Security requirements are not limited to access control but also include flow control to manage interactions between different components of the WFMS. If a piece of information Info flows from a role R_1 to another role R_2 of the same or different organization, we must be sure that R_2 has privilege allowing it to get an access to Info. If R_2 must not access to Info and R_1 transmits the information Info to R_2, a leakage of information occurs. The problem is known as the "confinement problem" [WEB96, WEB] and requires a flow control model to solve it. So, a workflow specification must be associated with a security policy which takes into account

* Telecom Bretagne 2, rue de la Chataigneraie, CS 17607, 35576, Cesson Sévigné Cedex France

access control and information flow control simultaneously. Since a workflow execution implies interaction between different entities, how to deploy a security policy becomes a critical issue. Either with a centralized or distributed workflow, we have to define different responsibilities of different workflow entities to enforce the whole security of the workflow execution.

Many research works have been interested in WFMS security but no study has addressed the deployment part. Several proposals [NAH98, VA96, PH99, EB99] suggest different access control models to manage security in WFMS without defining how to implement such a policy. They have only defined a centralized procedure that controls the execution of the workflow. [VA96, NAH98] propose a conceptual and a logical model WAM (Workflow Authorization Model) to enforce the authorization flow based on the inter-dependencies between tasks using colored and temporized petri nets. Hung and Karlapalem [PH99] have presented an authorization model for WFMS that addresses three security properties: integrity, authorization and availability. [EB99] studies authorizations in WFMS considering different constraints. In this work, Bertino describes the specification and enforcement of authorization constraints in workflow management systems. This work enumerates and defines possible configurations that satisfy the workflow execution and associated constraints. But it does not show how a manager must proceed to deploy the workflow without violating its constraints. Also, it does not consider the inter-organizational case where many organizations can interoperate to achieve a common objective. These works suppose the existence of a central entity responsible for specifying and managing the workflow security policy without showing how such a policy will be used. They do not deal with the communications required to apply the workflow security policy. Also, they do not show how this security policy can be dynamically supervised during the workflow execution.

In this paper, using an access and flow control based on the OrBAC model [SAC07], we present an approach to deploy a security policy either in intra or in inter workflows. This work actually focuses on how to manage the workflow security policy. It answers to the following questions: Who defines the workflow security policy? Who is responsible for maintaining a secure execution environment? Who has permission to manage the workflow security policy? The paper introduces PEP and PDP entities required to deploy the security policy within the workflow. We show how these entities have to communicate with workflow components in order to synchronize and manage the security policy. In [SAC08] a workflow security policy is modelled and specified jointly with the workflow specification. We show in this paper how to deploy a workflow policy and how it must be applied to different components of the workflow. The deployment depends necessarily on the execution mode and type of the workflow (intra or inter organizational workflow). Then we exemplify the inter-organizational case through the telesurgery application.

The paper is organized as follows. Section 2 introduces the workflow management systems and their intra or inter organizational types. Section 3 presents provisioning and outsourcing approaches to manage security policy within a workflow. Section 4 shows different WFMS policy requirements. Section 5 explains how to define the security policy associated to WFMS and how to use the OrBAC model for this purpose in both intra and inter organizational cases. Section 6 specifies our approach to deploy a security policy either in intra or inter organizational workflow. Then, in section 7, we exemplify the inter-organizational case through a telesurgery application. Section 8 concludes the paper and

outlines future work.

This paper is extended version of [SAC09], especially section 7 is new.

2 Workflow Management Systems

A workflow is a representation of a process. This process is divided into many tasks having inter-dependent executions. The execution of these different tasks may include temporal constraints or conditions. We can actually suppose different manners to constrain tasks execution: two tasks may be executed in sequential, parallel or concurrent mode. A workflow is composed of two phases: a building phase and an execution phase. The first phase defines the formal representation of the process. The second one is an instantiation of the workflow which is based on the specification defined in the first phase.

A WFMS is a system which supports the specification, control, coordination and administration of processes using workflows. This is done through the execution of a software which is based on the workflow structure. WFMSs are used in different domains: research, industry, commerce, etc. WFMSs may include sensitive resources and potentially malicious subjects when managing and executing different workflows. This introduces the need to care about security in these systems.

Tasks of a workflow may be executed by the same subject or by different ones having different roles that need to access to different resources of the system. Tasks must be executed by authorized subjects and according to their execution order specification. In addition, subjects must have access only to authorized objects during the time of task execution. Thus, granting or revoking privileges must be synchronized with the workflow execution progress. All these requirements lead us to combine the specification of the workflow and the security policy associated with it. This security policy must ensure many properties: integrity, authorization, availability, confidentiality, authentication and separation of duty.

2.1 Centralized and distributed workflows

A WFMS can be executed either in centralized or distributed modes. With a centralized control, there exists a single central WFMS which is responsible for (1) distributing the tasks to the appropriate agents, and (2) ensuring the task dependencies by sending the tasks to the respective agents only when all requisite conditions are satisfied. This entity has a whole view of the workflow. It is the initiator of the workflow and it evaluates execution results in order to choose the next agent to whom it will send the next task. In this mode, workflow agents do not know the dependencies between each other.

In the distributed execution mode the whole workflow is sent by the initiator to the first agent A_1. After the execution of the first task associated with the agent A_1, A_1 evaluates the dependencies with its successors. The rest of the workflow is then sent to the valid successor depending on the result of the first task. Finally the last agent sends the final result to the central entity which initiated the workflow. In this mode, workflow agents know the different dependencies with other agents. Especially, each agent knows its successors and conditions that are needed to transit from a task to another. Also different constraints defined with the workflow specification are ensured by different agents which have to check if they are respecting workflow constraints when executing different tasks. These two modes are represented in figure 1.

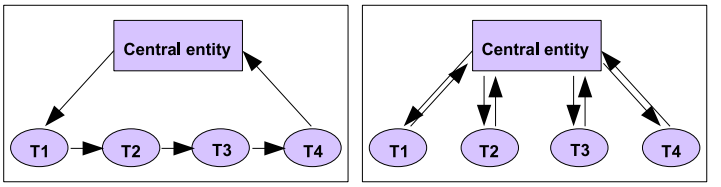

Fig. 1: Centralized and distributed workflow

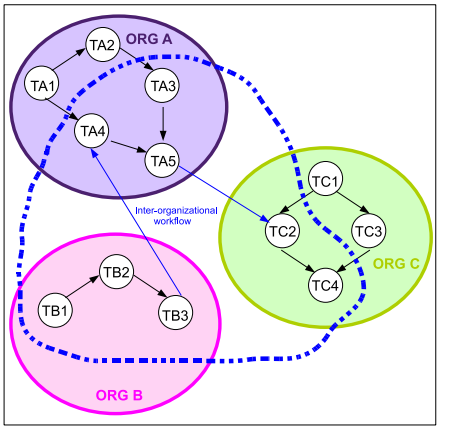

Fig. 2: Inter-organizational workflow

2.2 Intra and inter organizational workflows

Workflow tasks can be executed within the same organization or within different ones. We respectively call these cases intra and inter organizational workflows. Intra-organizational workflows are simpler to manage since all process requirements (resources, subjects...) belong to the same organization. In the inter-organizational case, specific requirements are needed. Figure 2 represents an inter-organizational workflow. Tasks of this workflow belong to three different organizations (ORGA, ORGB and ORGC). These tasks can belong or not at the same time to other local workflows in their own organizations. As a functional requirement, the execution of an inter organizational workflow needs synchronization between the execution of different sub-workflows that participate in the collaboration. As a security requirement, we have to deal with interactions between these different organizations. The execution of an intra-organizational workflow can be either centralized or distributed as it is explained in section 2.1. The execution of inter-organizational workflows can be considered distributed since each sub-workflow is executed within an independent organization. But the coordination between these different organizations can be done either by these organizations themselves (distributed mode) or through a central entity which deals with the synchronization and the orchestration of different sub-workflows (centralized mode).

3 WFMS security policy management

Functionally, the central entity of a workflow can be either a coordinator entity which controls the whole execution of the workflow in a centralized workflow or just an initiator entity in a distributed workflow. This entity will be also responsible for the security of the workflow. For this aspect, a choice must be done by the workflow administrator, i.e. the security requirements may be managed either using an outsourcing approach or a provisioning approach.

3.1 Outsourcing approach

In this approach, the central entity of the workflow is also responsible for the workflow security. Figure 3 represents this approach within a centralized and distributed workflow. Arrows in this figure show different communications during the workflow execution. We notice that access to the security policy is only done by the WFC (Workflow Coordinator). Using this approach, workflow agents have to ask for permissions to execute workflow tasks to the central entity. This central entity has to check if the agent is authorized to execute the task depending on the workflow security policy and the workflow specification which includes workflow constraints. Even if the execution of the workflow is distributed, the central entity will need all information required about the execution progression when it is asked for a permission by workflow agents. The security policy is managed by the central entity. The different worklow agents do not have any knowledge about it. They have just the obligation to ask for a permission to execute requested workflow tasks. The central entity synchronizes the security policy with the execution progression of the workflow.

3.2 Provisioning approach

In the provisioning approach, the central entity delegates the workflow security requirements to different agents. So, if an agent needs to execute a workflow task it will not ask permission to the central entity but it will search it from a security server. This check is done by all workflow agents. This approach reduces the bulk on the central entity but it increases the execution time on the workflow agents. Figure 4 represents this approach with a centralized and a distributed workflow. Arrows in this figure show different communications during the workflow execution. We notice that access to the security policy is done by workflow agents. With this approach the central entity has no security functionalities. It does not deal with the security part which is managed by agents.

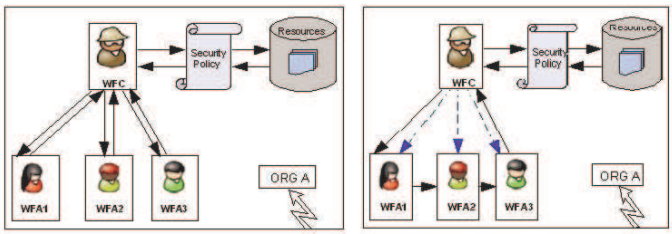

Fig. 3: Outsourcing in a centralized and a distributed workflow

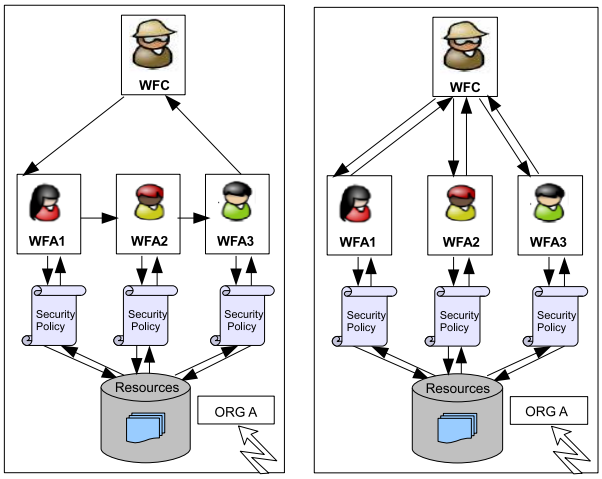

Fig. 4: Provisioning in a centralized and a distributed workflow

4 WFMS security policy requirements

4.1 Workflow constraints

A workflow specification is correlatively related to a set of constraints which complete the workflow definition. These constraints can concern either workflow tasks or subjects or roles that execute these tasks. We classify workflow constraints into two classes: static constraints and dynamic constraints.

Static constraints This type of constraints can be evaluated during the user role assignment time. This type includes the separation and binding of duties principles.
Definition 1: SC is the set of static constraints of a workflow. SC is composed of atoms and rules. If T_i, T_j and T_k are workflow tasks and S_i is a subject belonging to the workflow and c_i is a workflow constraint then:
$c_i \in \text{SC} \Leftrightarrow c_i \in \{\text{same\_subject}(T_i, T_j), \text{different\_subject}(T_i, T_j), \text{must\_execute}(S_i, T_j), \text{execute}(S_i, T_j) \rightarrow \neg\text{must\_execute}(S_i, T_k)\}$

The interpretation of this set of constraints is given below:

- Tasks T_i and T_j must be executed by the same subject
- Tasks T_i and T_j must be executed by different subjects
- Task T_j must be executed by subject S_i
- If S_i executes task T_j then it must not execute task T_k

Dynamic constraints Dynamic constraints have to be evaluated during the execution time. Once the workflow execution starts, the workflow manager has to survey it in order to control the consistency of dynamic constraints. This type of constraints includes essentially temporal constraints.

Definition 2: TC is the set of temporal constraints of a workflow. TC is composed of atoms and rules. If T_i, T_j and Tk are workflow tasks and c_i is a workflow constraint then: $c_i \in \text{TC} \Leftrightarrow c_i \in \{$ Start_Before(T_i, T_j), Start_With(T_i, T_j), End_Before(T_i, T_j), End_With(T_i, T_j) $\}$

The interpretation of these constraints is:

- Task T_i must start before task T_j
- Tasks T_i and T_j must start at the same time
- Tasks T_i must end before task T_j
- Tasks T_i and T_j must end at the same time

Another type of these constraints results from the workflow specification itself. A conditional execution in the workflow specification is interpreted as a dynamic constraint. An example of these constraints is: If T_i has been executed then execute T_j otherwise, execute T_k.

4.2 OrBAC in brief

To define a WFMS policy and its secure execution environment, we suggest using the OrBAC model. Thus, before presenting how to manage the security policy associated with the workflow execution, we first briefly recall basic concepts suggested in the OrBAC model.

In order to specify a security policy, the OrBAC model [AAeKT03, FCM04b] defines several entities and relationships. It first introduces the concept of organization which is central in OrBAC. An organization is any active entity that is responsible for managing a security policy. Each organization can define its proper policy using OrBAC. Then, instead of modelling the policy by using the concrete implementation-related concepts of subject, action and object, the OrBAC model suggests reasoning with the roles that subjects, actions or objects are assigned to in an organization. The role of a subject is simply called a role as in the RBAC model. The role of an action is called activity and the role of an object is called view. Each organization can then define security rules which specify that some roles are permitted or prohibited to carry out some activities on some views. Moreover, an organization can be composed of several sub organizations, each one having its own policy. It is also possible to define a generic security policy in the root organization. Its sub organizations will inherit from its security policy. Also, they can add or delete some rules and so, define their proper policy. The definition of an organization and the hierarchy of its sub organizations facilitate the administration [FCM04a]. The security rules do not apply statically but their activation may depend on contextual conditions [CCB]. For this purpose, the concept of context is explicitly included in OrBAC. Contexts are used to express different types of extra conditions or constraints that control activation of rules expressed in the access control policy. So, using a formalism based on first order logic, security rules are specified using a 6-places predicate.

Definition 3: an OrBAC security rule is defined as: security_rule (type, organization, role, activity, view, context) where type $\in$ {permission, prohibition, obligation}. An example of this security rule can be: security_rule (permission, a_hosp, nurse, consult,

medical_record, urgency) meaning that, in organization a_hosp, a nurse is permitted to consult a medical record in the context of urgency.

From an abstract policy, we can then derive a concrete policy that applies to subject, action and object using the following OrBAC predicates:

- empower(org, subject, role): means that in organization org, subject is assigned to role.
- consider(org, action, activity): means that in organization org, action is considered an implementation of activity.
- use(org, object, view): means that in organization org, object is used in view.

5 WFMS security policy specification

5.1 Intra-organizational workflows

To specify the workflow security policy we use the OrBAC model. Using constraints defined with the workflow specification, the security server defines the different contexts of security rules. These contexts make the security policy dynamic and synchronized with the workflow execution. Workflow constraints are either static or dynamic as shown in section 4.1. In OrBAC, there are built-in contexts to define such constraints. *Prerequisite context* aims to restrict or extend privileges granted to a role depending on some conditions. So, this context category is useful in WFMS to specify constraints associated with the workflow process execution. The provisional context depends on previous actions the subject has performed in the system, i.e. it corresponds to a history of execution. Provisional contexts are very interesting in the domain of WFMS since the execution of a task depends on the history of execution of precedent tasks. Also, it permits the definition of a dynamic security policy according to contexts, a very useful requirement in WFMS. Using the workflow specification the policy server specifies an abstract policy of the workflow. Such a policy is based on roles, activities, views and contexts. Until now, the policy does not deal with the instantiation of the workflow. This policy is defined within the organization where the workflow process will be executed.

5.2 Inter-organizational workflows

For inter-organizational workflows, several workflows are involved in the achievement of a common objective. Interactions between different organizations are necessary in this case. Such communications have to be well controlled since each organization has its private information, its private policy and its own characteristics. A communication protocol is needed in order to ensure these different communications and flows that can transit from an organization to another. This requirement is critical especially to enforce the constraints defined with the workflow specification (see section 4.1).

If these constraints are defined within the same organization, the WFC can enforce and check if its user role assignment violates the base of constraints or not. Otherwise, if we suppose for example a constraint that specifies that task T_i must start before task T_j and that task T_i is provided by an organization OrgA and task T_j is provided by an organization OrgB, enforcing workflow constraints becomes more complicated. Indeed,

synchronization of tasks between different organizations must be done in order to enforce constraints.

5.2.1 Creation of VO

For both types of constraints, we need to create a virtual organization (VO) in order to synchronize execution of tasks between different organizations and to enforce different workflow constraints. The VO is responsible for the secure inter-organizational communications. To create a VO, the inter-organizational workflow coordinator (IWFC) sends requests to different organizations participating in the global workflow. Receiving these requests inviting them to join the VO, WFCs have to answer the IWFC to inform it about their decision to participate in the VO. Once the agreement is achieved, the WFCs must communicate to the IWFC information about their local workflow. They must especially inform the IWFC about the constraint base. This piece of information includes the user task and the role user assignment that the WFC had defined within its organization. So, each organization orgI sends to the IWFC a set defined as: $\{(T_i,\ R_i,\ U_i),\ IC_i\}$. The first triple means that the user U_i is assigned to the role R_i to execute the task T_i. The second argument is the set of inter-organizational constraints defined within the organization orgI. This set includes constraints defined within the organization orgI and having dependencies with other organizations. Receiving these pieces of information, the IWFC will have a global vision of all organizations. So, the IWFC has to check the consistency of global workflow constraints. The IWFC verifies if the different assignments received from different organizations violate or note the set of inter-organizational constraints of the global workflow. If an inconsistency is detected, the IWFC transmits the information to organizations concerned with this inconsistency. These organizations have to re-assign a new user to the role and task causing the conflict. Then this new assignment is sent to the IWFC who checks again the inter-organizational set of constraints. These communications between the IWFC and different organizations continue until obtaining a consistent inter-organizational constraints base.

5.2.2 VO security policy

Based on the different constraints that the IWFC gathers, it has to define a coordination security policy and to administrate it in order to ensure the non violation of the constraints during the workflow execution. To define its policy, the IWFC must transform the constraints base into a set of security rules. These security rules constitute the inter-organizational policy of the global workflow. To communicate to each other, organizations have to be controlled by this policy. Most workflow constraints are based on tasks of the workflow and subjects executing these tasks. So, the VO has to manage the transition of subjects from an organization to another. The VO security policy must be consistent with the different local security policies of organizations. Also, it must have a higher priority than local policies.

For example, let us consider the following constraint: "Task T_2 of organization OrgA and task T_4 of organization OrgB must be executed by the same subject." The IWFC can receive this constraint either from the OrgA or from the OrgB. These organizations know about the inter-organizational constraints that they must satisfy but they do not know how to ensure such a type of constraint. Indeed, each organization does not have the security policy of the other organization. So, a third part, which is the VO, is responsible

for managing this communication. The subject who will execute the task T_2 into the organization OrgA can have a different role when executing the task T_4. We suppose that Alice is the subject who will execute these tasks. Thus, for these two tasks, OrgA and OrgB can respectively define these rules and role assignments in their local security policies:
security_rule(permission, OrgA, clerk, sign, check, nominal).
empower (OrgA, Alice, clerk).
security_rule(permission, OrgB, manager, validate, mission, nominal).
empower (OrgB, Alice, manager):-
empower (OrgA, Alice, clerk),
security_rule(permission,OrgA, clerk, sign, check, nominal).

To be sure that the subject Alice is executing the task T_2 of OrgA, the WFC_2 has to inform the IWFC about the execution progress of its workflow. Such a communication allows the IWFC to update its security policy and its different contexts. Indeed, if we suppose that when the task T_2 will be executed, Alice will not be ready to perform the task, the WFC_2 will be obliged to change the subject executing the task and must inform the IWFC about this change.

So, to enforce the constraint above, Alice has to switch from the role clerk within the organization OrgA to the role manager within the organization OrgB. This transition may imply a leakage of information. The problem is more dangerous if Alice transfers a confidential or sensitive data. This confinement problem has to be managed using an information flow control policy managed by the VO. To define this policy, we are based on the information flow control model DTE (Domain and Type Enforcement) [LB95, KW03]. An integrated model based on OrBAC model and using a DTE approach is presented in [SAC07]. Let us define the entities that this model introduces.

Definition 4: (domain) S is a set of all system subjects (active entities). S is divided into equivalence classes. Each class represents a domain D including a set of subjects having the same role in the system.

Definition 5: (type) O is a set of all system objects (passive entities). O is divided into equivalence classes. Each class represents a type T including a set of objects having the same integrity properties in the system.

Definition 6: (Entry Point) An entry point is a program or an activity which must be executed to pass from a domain D_1 to a domain D_2, denoted EP(D_1, D_2) or $EP_{1,2}$. An entry point implies two rules: (1) subjects passing from D_1 to D_2 obtain a set of privileges depending on the entry point they execute, (2) subjects passing from D_1 to D_2 lose all their D_1 privileges.

The first rule means that an entry point defines the set of privileges that subjects will obtain when moving from a domain D_1 to a domain D_2. These privileges are included into or equal to the set of privileges that D_2 subjects have. Each domain can have more than one entry point. The execution of these different entry points implies different privilege sets. The second rule means that if a subject leaves a domain it cannot return to it only by executing one of its entry points.

To be consistent with the OrBAC model, a domain corresponds, within an organization, to a role. The entry points to a domain correspond to different roles assigned temporarily to a subject which does not belong to the organization where this role is defined. Thus,

to define its security policy, the IWFC sends a request to different organizations to get their entry points to their different roles (i.e. domains). Then the IWFC security policy is a set of rules defined as: security_rule(permission, VO, D_i, Enter, D_j, through(E_i, j))

IWFC rules are specified as specific OrBAC rules. The rule above expresses the transition between the domain D_i belonging to an organization Org_i to the domain D_j belonging to an organization Org_j and uses the entry point concept to define a new context in order to preserve secure information flows and to keep DTE concepts. Therefore, to consider this particular rule we suppose the following hypotheses: (1) the source domain is considered a role in the OrBAC rule, (2) the destination domain is considered a view in the OrBAC rule, (3) the transition between two domains can be expressed as an OrBAC activity since the basic meaning of this specific rule is to handle interactions between domains. For this purpose, we define the OrBAC Enter activity, (4) the entry point defines the manner to enter into a domain. Thus, an entry point can be included into a specific context in an OrBAC rule denoted Through($E_{i,j}$). This context specifies that the rule is valid only through the $E_{i,j}$ execution.

Through(E_i, j) context it is a composed context defined by an organization Org_j as Through($E_{i,j}$) = {$E_{i,j}$, Pr_j, coming_from(Org_i)}. The first argument represents the activity to execute to enter a domain. The second argument gives the set of privileges that will be granted to the subject who intends to enter a domain D_j when coming from a domain D_i. This set of privileges depends on the activity executed to enter the domain. Since the entry points can also depend on different organizations from where subjects are coming, we define the last argument coming_from(Org_i). For example, let us suppose that we have two organizations Org_j and Org_k which define a role secretary. If two subjects having this role and coming respectively from Org_j and Org_k need to move to a role clerk in the organization Org_i, they will not get the same set of privileges even if they execute the same entry point. This results from different sensitivity degrees of different organizations and also from the relationship between the organizations. For instance a special privilege can be granted to subjects coming from an organization and moving to an organization with which some agreement has been established. This context can be composed with other OrBAC contexts. Then it may include extra conditions and circumstances to supervise the flow control.

The communication between different WFCs and the IWFC are considered secure and all organizations are considered belonging to the same trust alliance. If it was not the case, the policy of the IWFC would be much more complicated and more dynamic. All communications have to transit through a trust server to communicate information about actual status and about the local worflows progress.

6 Deploying WFMS security policy

With the diversity of resources, roles and tasks in a workflow, the deployment of a WFMS security policy becomes a critical issue. In this section we show how to manage such a security policy either in an intra or inter organizational workflow. The second case is more complex since it deals with different communications between organizations cooperating to execute the global workflow. We use OrBAC to specify our security policy. As shown above, this model is able to represent confinement aspect using the organization concept.

6.1 Intra-organizational case

The deployment of the security policy depends on the execution mode of the workflow and on the approach that we are using to manage the policy. For the intra-organizational case, we suppose that we deal with a distributed workflow. In a centralized workflow, all communications in the workflow are done through the workflow coordinator (WFC). Figure 5 represents a general scheme for deploying a WFMS security policy using the outsourcing approach. The explanation in this section follows the numbers on the arrows in the figure.

6.1.1 Specification environment

As shown in figure 5, the general scheme is divided into a specification environment and an enactment environment.

(1) The definition of the workflow specification takes place. The process designer provides a written or a graphic specification of the different tasks of the workflow and of the roles that have to execute these tasks. Also, the designer defines different constraints associated with the workflow definition.

(2) Different specification components are transmitted to a modelling tool. Petri nets are an efficient tool to model workflows. Aside representing the workflow evolution, Petri nets can take into account temporal constraints that may be defined with the workflow. With Petri nets, it is possible to specify that the execution of activities must be done in sequence, or in parallel, or that a choice has to be made between activities (alternative activities), or that certain activities need to be executed more than once (iterative activities). An approach using Petri nets to represent workflows is explained in [SAC08].

(3) Once the workflow is modelled, the process modelling tool communicates with a process verification engine to check if the workflow is well formed. If the workflow is modelled using Petri nets, there are Petri nets properties (correctness, liveness, well-structuredness, soundness, ...) that must hold to conclude that the workflow is well defined and specified.

(4) Once the workflow is well formed, workflow constraints are checked. Constraints validation engine checks the consistency of constraints defined with the workflow specification.

(5) The policy server has to get information about the workflow specification, so that the security policy can be specified. Thus, the process modelling tool transmits workflow information (roles, tasks, objects, constraints) to the server policy. Based on these elements, the policy server specifies the workflow security policy to be used to ensure a secured execution of the workflow.

(6) The OrBAC API creates the workflow security policy and saves it into a policy repository (PR). This policy is an abstract policy which is based on abstract entities defined in the OrBAC model. From this abstract policy saved in the PR, a concrete policy will be derived when the workflow is instantiated. This policy is defined with non active contexts. These contexts contain initial constraints defined with the workflow specification. With the progression of the workflow execution, these contexts are updated and activated in order to make the security rules applicable. The process modelling tool provides the workflow specification to the process enactment engine. Then we move to the enactment environment.

6.1.2 Enactment environment

It manages the workflow instantiation. In this environment the workflow specification moves to the run time phase. The user assignment is done at this phase depending on the workflow specification. The steps below define the next phases to be done once the workflow specification is ready. In this section, we actually present a decentralized approach to execute the workflow. Regarding the security part, the outsourcing approach is used. So, the WFC is responsible for the security of the workflow execution. To manage the security policy, the WFC needs a PEP (Policy Enforcement Point) in order to deal with different policy instances. This logical entity is responsible for controlling access to different system resources. Before starting the workflow execution, the WFC has to know how to manage the workflow security policy. So, communications with the policy server are initiated. On the side of the policy server, there is a PDP (Policy Decision Point) who is responsible for communicating with the security client and for replying to its different requests. When a user tries to access a file or other resource on a computer network or server that uses policy-based access management, the PEP will describe the user's attributes to other entities in the system. The PEP will ask the PDP whether or not to authorize the user based on the description of the user's attributes.

(7) In the process enactment engine, the WFC entity deals firstly with the assignment of users to different roles and tasks. This assignment is based on a communication with different users and external applications that can participate to the workflow execution.

(8) The assignment made by the WFC must be checked with respect to the workflow constraints. Such an assignment must not make the constraint base inconsistent. So the process enactment engine has to communicate with the constraints validation engine to validate its assignment and to apply it.

(9) The PEP sends a request to the PDP. The request must contain information about the assignment of users to different tasks and the different contexts of the workflow execution. Then, the PDP receives the request. This logical entity or place on a server makes admission control and policy decisions in response to a request to access system resources.

(10) After receiving concrete entities and information about the workflow execution, the PDP communicates with the PIE (Policy Instantiation Engine) to ask for the concrete policy.

(11) The PIE is responsible for generating a concrete security policy based on information received from the PDP and the abstract policy saved in the PR. The PIE updates contexts and transfers the new security policy to the PDP.

(12) The PDP takes the policy decision. Whenever it receives a new policy instance, i.e. a concrete rule (permission or prohibition), a PDP has to map this piece of information onto concrete actions to push the new policy into the PEPs. Thus, the PDP has to be aware of the PEPs capabilities, so that it can translate the security rules into generic configurations and then the generic configurations into specific configurations, considering the implementation of the PEP. So, the PDP translates the concrete policy into a translatable language and stores it into a PIB (Policy Information Base). Applicable policies are stored on the system and are analyzed by the PDP.

(13) The PDP makes and returns the decision. The PEP will let the user know whether or not he or she has been authorized to access the requested resource.

(14) The PEP sends an acknowledgment to the PDP to inform about its acceptance of the PDP decision and how it will apply it.

(15) Once the workflow specification and a consistent policy has been defined, the WFC initiates the workflow. It sends the whole specification to the first agent that will execute the first task of the workflow. Since this workflow execution is distributed, each agent responsible for executing a task, sends a request to the WFC in order to grant him the authorization to accomplish the task. These requests are represented by blue arrows in figure 5. When the WFC receives such a request, it takes the decision to accept the accesses. Then, it sends a response to the workflow agent including the permission or prohibition to execute the task. After the first task, agents have to send the execution context to the WFC. This context provides information about the progression of the workflow. The WFC needs such information to be able to update its policy. In fact, when a request is received from a workflow agent, the WFC has to communicate with the PDP and steps 9-14 are executed for each communication between the WFC and workflow agents. The execution context especially indicates about previous tasks that have been accomplished. If we consider for example a workflow where a task T_4 can be executed only if task T_1 has been executed successfully and then tasks T_2 and T_3 have been executed in parallel and successfully, then the authorization is granted to agent executing T_4 only if the execution context holds and the WFC is informed about this.

Steps (16) and (17) show the functional communication between workflow agents. Blue arrows show the requests for the authorization rules from the WFC.

(18) The last agent sends the workflow result to the WFC, and so the workflow execution completes.

If we consider that the execution of the workflow is centralized, the communication between WFC and workflow agents to ask for authorizations will not exist. The WFC will actually send the task and the authorization associated with the task execution to the workflow agent and the agent will respond with the execution result. By receiving this result the WFC will update its security policy and select the next agent depending on the result of the task execution that he has received.

If we suppose that the security policy is managed using the provisioning approach, then the WFC will have no idea about the security policy to be deployed during the workflow execution. Also, workflow agents will communicate directly with the policy server to ask for an authorization to execute their tasks. Thus, each workflow agent will have a Local PEP (LPEP) and these LPEPs will communicate with the PDP present on the policy server. In addition to information about their capabilities, LPEP have to communicate to the PDP information about the workflow execution progress.

6.2 Inter-organizational case

With the inter-organizational case the same approach can be applied. The specification environment still unchanged. In the enactment environment, communications are more complicated. The WFC is replaced by the IWFC for the inter-organizational coordination entity and each WFA is replaced by the WFC of each organization. Then, different communications between organizations are done as described in section 5.2. Once the VO is well formed, the IWFC of this VO communicates through its PEP with the PDP of the policy server to request the policy decision. With the outsourcing approach this entity will maintain the security of the inter-organizational workflow during its execution. If we are in a provisioning approach, each organization has to implement its local PEP (LPEP). This LPEP is used to communicate directly with the policy server without passing by the

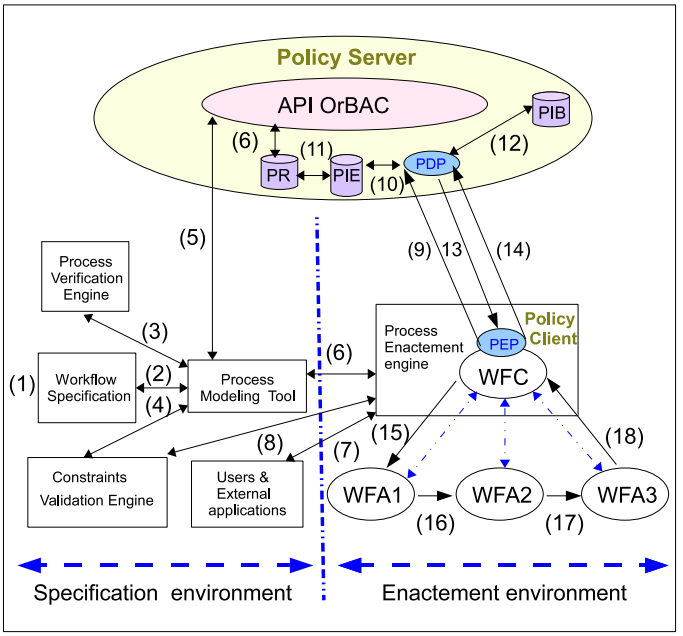

Fig. 5: Security policy deployment using an outsourcing approach

VO. This virtual organization will be useful just for the functional part of the workflow. The next section section exemplifies the inter-organizational case to show the evolution of the security policy during the workflow execution.

7 Case study: telesurgery

Telesurgery can be considered an interesting inter-organizational workflow. The application includes at least two independent organizations; i.e. two hospitals, which have to collaborate in order to achieve a common goal. This goal is generally to operate successfully a patient. This application has to enforce precise temporal constraints when it is executed. These constraints consist especially on the very precise parallelism requirements to be enforced between the two sites connected to the application. In this example, we suppose that we have a telesurgery operation between the hospital of New York and the hospital of Strasbourg. The patient is located in the hospital of New York. This organization is composed of two sub_organizations: the robotic team and the medical team. In the robotic team, the robot and a set of technicians will act. In the medical team, assistants will survey the evolution of the patient state. The remote doctor is located in Strasbourg hospital. This doctor will manage a surgical console which controls the move of robot's arms in the sub_organization "robotic team". The reproduction of the doctor actions must be done with accuracy in real time by the robots's arms. In the following, we define the set of different elements of the application. The figure 6 shows a representation of a simple and general workflow of principal activities to do during a telesurgery operation. The two states *"S"* and *"E"* represent respectively the initial and the end states of

the workflow.

- Organizations = {Strasbourg_hospital, NewYork_hospital}
- Sub_organizations(NewYork_hospital) = {robotics_team, medical_team}
- Roles = {surgeon, assistant, technician, anaesthetist, robot}
- Views = {surgical_console, patient, medical_record, robot's arms, video_display_terminal}
- Contexts = {bleeding, reaction_to_anaesthetic (RA), non_reaction_to_anaesthetic (NRA)}
- Activities = {T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}}

where T_1 = Initialize robot's arms, T_2 = Create a working zone, T_3 = Command the surgical console, T_4 = Reproduce the doctor actions on the console, T_5 = Stop the action of robot's arms, T_6 = Reanimate the patient, T_7 = Supervise vital parameters of the patient, T_8 = Update the online information of the patient, T_9 = Carry over the operation, T_{10} = Supervise robot's actions.

In this example we are especially interested in the enactment environment of figure 5. We suppose that our workflow is well formed and that all constraints are not violated. Numbers used in this section refer to different rows defined above in figure 5. In step 5, the policy server will receive information about the workflow to specify its security policy. This information is the assignment of different roles to different tasks within different organizations. For this purpose, we use the function *assignment(Org)* to specify the set of assignments done within Org during the workflow specification. This set is composed of a quadruple (T, R, O, C) meaning that the task T has to be executed by the role R using the object O and satisfying the conditions C. If there is no condition this field will be empty represented by "-".. We can consider also that Org can be a sub_organization.

- assignment(Strasbourg_hospital) = {(T_1, surgeon, surgical_console, -), (T_2, surgeon, surgical_console, -), (T_3, surgeon, surgical_console, -)}
- assignment(robotic_team) = {(T_4, robot, robot's arms, -), (T_5, technician, robot's arms, bleeding), (T_6, anaesthetist, patient, R.A)}
- assignment(medical_team) = {(T_7, assistant, video_display_terminal, -), (T_8, assistant, medical_record, N.R.A), (T_9, assistant, medical_record, -), (T_{10}, assistant, video_display_terminal, -)}

Using these assignments about the workflow specification, the policy server specifies its abstract security policy using the API OrBAC. This policy describes permissions to execute the different workflow tasks. So, the set of assignment is translated to the following *abstract security policy* (SP):
SP = {R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}}

- R_1 = (permission, Strasbourg_hospital, surgeon, T_1, surgical_console, default)
- R_2 = (permission, Strasbourg_hospital, surgeon, T_2, surgical_console, default)

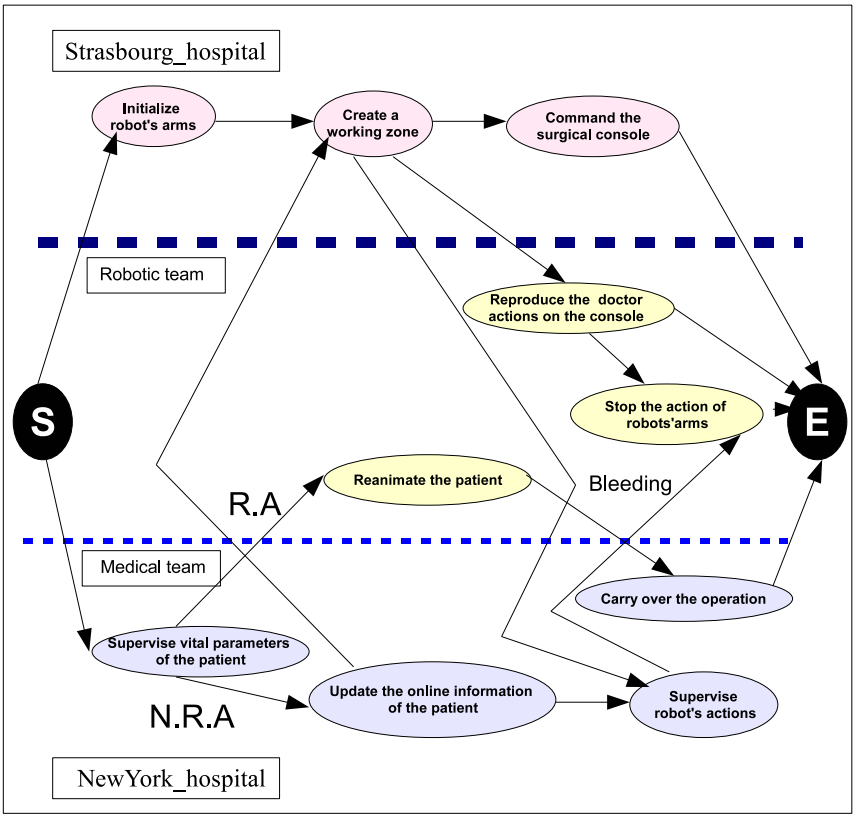

Fig. 6: A simple workflow of a telesurgery operation

- R_3 = (permission, Strasbourg_hospital, surgeon, T_3, surgical_console, default)
- R_4 = (permission, robotic_team, robot, T_4, robot's arms, default)
- R_5 = (permission, robotic_team, technician, T_5, robot's arms, "bleeding")
- R_6 = (permission, robotic_team, anaesthetist, T_6, patient, "R.A")
- R_7 = (permission, medical_team, assistant, T_7, video_display_terminal, default)
- R_8 = (permission, medical_team, assistant, T_8, medical_record, "N.R.A")
- R_9 = (permission, medical_team, assistant, T_9, medical_record, default)
- R_{10} = (permission, medical_team, assistant, T_{10}, video_display_terminal, default)

In this policy we use the context "default" when there is no condition to satisfy when executing the task. Also, we suppose in this case that the same subject executing the task T_7 must execute the task T_6 (same_subject(T_6, T_7)). We notice that the Task T_6 is

executed within the sub-organization robotic_team and the Task T_7 is executed within the sub-organization medical_team. The order of execution of different tasks is represented in figure 6. When the process enactment engine receives the workflow specification it starts its user-role assignment for the execution of the workflow. In the following, we define subjects that we assign to different roles:

- empower(Strasbourg_hospital, John, surgeon)
- empower(robotic_team, OurRobot, robot)
- empower(robotic_team, Alice, anaesthetist)
- empower(medical_team, Alice, assistant)

The PEP of the IWFC communicates with the PDP of the policy server to tell him about the assignment that it has done. Combining this assignment with the abstract security policy stored into the Policy Repository, the PIE generates the concrete policy. This policy contains security rules defined with a default context, i.e. the set of permissions to execute the workflow under normal and usual circumstances. This security policy is defined as a set of *concrete rules*:
SP = {R_1', R_2', R_3', R_4', R_7', R_8', R_{10}'}

- R_1' = (permission, Strasbourg_hospital, John, T_1, surgical_console, default)
- R_2' = (permission, Strasbourg_hospital, John, T_2, surgical_console, default)
- R_3' = (permission, Strasbourg_hospital, John, T_3, surgical_console, default)
- R_4' = (permission, robotic_team, OurRobot, T_4, robot's arms, default)
- R_7' = (permission, medical_team, Alice, T_7, video_display_terminal, default)
- R_8' = (permission, medical_team, Alice, T_8, medical_record, "N.R.A")
- R_{10}' = (permission, medical_team, Alice, T_{10}, video_display_terminal, default)

This security policy is managed by the IWFC. When the execution of the workflow starts, each WFC must require the permissions to execute its different activities. This request is done at the execution of each task of the workflow. Let us suppose that during the execution of the workflow the context "R.A" is activated. In this case, the IWFC is informed of this new event within the workflow. This entity must communicate with the PDP of the server of the security policy. To generate the rule associated with this event the PIE activates the abstract rule R_6 defined under this context and other rules of permitted activities related to this condition. It generates the following concrete rules R_6' with the active context "R.A" and the rule R_9'. These rules are sent to the IWFC's PEP. It will relay them to the appropriate organization.

- R_6' = (permission, robotic_team, Alice, T_6, patient, "R.A")
- R_9' = (permission, medical_team, Alice, T_9, medical_record, default)

We can notice that the rule R_6' implies an information flow control that the IWFC has to manage. The subject Alice acting on the medical_team has to move to the robotic_team when the context "R.A" is activated. This subject has the role assistant in the medical_team organization and the role of anaesthetist in the robotic_team sub_organization. The IWFC defines an information flow security rule to manage this transition as explained in section 5.2.2. If we suppose that, to access this organization, Alice has to clock in we can express this rule as follow: **SR = (permission, VO, medical_team, Enter, robotic_team, (clock_in, (reanimate, patient), coming_from(medical_team)))**. This rule says that Alice will have only the permission to reanimate the patient, when she moves to the robotic_team.

8 Conclusion

Several works have investigated security in WFMS since these systems present special requirements of access and information flow control. They are characterized by a diversity of tasks, roles, subjects and objects. But once the workflow security policy is defined and specified, the workflow manager has to know how to deploy such a policy. In this paper, we address the new issue of deploying a workflow security policy. We present different approaches and modes to administrate the functional and the security part of a workflow. Then, we introduce an approach to deploy the workflow security policy, once it is defined and specified. This management is studied for two different cases: intra and inter organizational workflows. For the first case we have defined functionalities that different entities (WFC, PEP and PDP) must manage. This case is implemented through calls to the OrBAC API which is responsible for defining the workflow security policy. For the second case a more rigorous control is needed since communications between independent organizations are required. Thus, a virtual organization must be created to control the synchronization and the communication between different organizations. The VO is responsible for ensuring the non violation of workflow constraints and managing the transition between domains of workflow subjects. In both cases, the security policy is based on the OrBAC model to specify the workflow security policy. The last section exemplifies the inter-organizational case through the telesurgery application. In this work we assume that organizations involved in the workflow belong to some trust alliance that provides some degree of confidence between the different parts of an inter-organizational workflow. If a trust alliance does not exist we have to resort to a trust server who will be responsible for controlling the whole workflow execution. In a forthcoming paper, we intend to go in details of the implementation of the intra and inter organizational case.
Acknowledgements: This work was supported by the ANR-06-SETIN-012 project Polux.

References

[AAeKT03] P. Balbiani S. Benferhat F. Cuppens Y. Deswarte A. Miège C. Saurel A. Abou el Kalam, R. El Baida and G. Trouessin. Organization based access control. *IEEE 4th International Workshop on Policies for Distributed Systems and Networks. Lake Come, Italy*, 2003.

[CCB] F. Cuppens and N. Cuppens-Boulahia. Modelling contextual security policies. *International Journal of Information Systems*, vol 7(4).

[EB99] V. Alturi E. Bertino, E. Ferrari. The specification and enforcement of authorization constraints in workflow management systems. *ACM Transactions on Information and System Security*, 1999.

[FCM04a] N. Cuppens-Boulahia F. Cuppens and A. Miège. Inheritance hierarchies in the or-bac model and application in a network environment. *FCS'04, Turku, Finlande*, 2004.

[FCM04b] T. Sans F. Cuppens, N. Cuppens-Boulahia and A. Miège. A formal approach to specify and deploy a network security policy. *Second FAST Workshop, Toulouse, France*, 2004.

[KW03] Jan Kiszka and Bernardo Wagner. Domain and type enforcement for real-time operating systems. *ETFA'03, Emerging Technologies and Factory Automation*, 2003.

[LB95] David L. Sherman Kenneth M. Walker Sheila A. Haghighat Lee Badger, Daniel F. Sterne. Practical domain and type enforcement for unix. *IEEE Symposium on Security and Privacy, Oakland, CA, USA*, 1995.

[NAH98] V. Atluri N.R. Adam and W-K. Huang. Modeling and analysis of workflows using petri nets. *Journal of Intelligent Information Systems, Special Issue on Workflow and Process Management*, 1998.

[PH99] Karlapalem P. Hung. A secure workflow model. *AISW (Australian Information Security Workshop)*, 1999.

[SAC07] Nora Cuppens-Boulahia Samiha Ayed and Frédéric Cuppens. An integrated model for access control and information flow requirements. *Proceedings of ASIAN'07, Doha, Qatar*, 2007.

[SAC08] Nora Cuppens-Boulahia Samiha Ayed and Frédéric Cuppens. Deploying access control in distributed workflow. *Sixth AISC Conference, Wollongong, NSW, Australia*, 81, 2008.

[SAC09] Nora Cuppens-Boulahia Samiha Ayed and Frédéric Cuppens. Deploying security policy in intra and inter workflow management systems. *Forth ARES Conference, Fukuoka, Japan*, 2009.

[VA96] W. Huang V. Atluri. An authorization model for workflows. *Proceedings of the Fifth European Symposium on Research in Computer Security, Rome, Italy*, 1996.

[WEB] W. D. Young W. E. Boebert, R. Y. Kain. The extended access matrix model of computer security. *ACM Sigsoft Software Engineering Notes*, 10(4).

[WEB96] Richard Y. Kain William E. Boebert. A further note on the confinment problem. *IEEE 1996 International Carnahan Conference on Security Technology, New York*, 1996.