Constraint-based Authorization Management for Mobile Collaboration Services

Yuqing SUN
Shandong University
School of Computer Science and Technology
sun_yuqing@sdu.edu.cn

Matthias FARWICK
University of Innsbruck
Institute of Computer Science
matthias.farwick@student.uibk.ac.at

Dickson K.W. CHIU
Dickson Computer Systems
7 Victory Avenue, Kowloon, Hong Kong
dicksonchiu@ieee.org

Abstract

With the fast development of high speed wireless technologies and the growing population of mobile portable devices, location information is potentially available for access control systems. Such applications are especially meaningful in emergency situations, where quick responses are urgently required for persons to be physically present in a certain place to perform sensitive tasks without conflicting with security policies. In this paper, we investigate this challenging problem and propose a novel Constraint-based Authorization Management Model, which takes the mobile execution of tasks with handheld devices into account. The authorizations are activated by means of Location Based Execution Binding to handle uncertain conditions such as flexible business processes and emergency situations, considering both the user's location and attributes. With the introduced algorithms the model is capable of execution planning to detect and avoid inconsistencies in the security constraints of activities at design and runtime. Finally we propose a system architecture based on Web service technologies and a XACML based syntax for defining the security constraints.

1 Introduction and Motivating Example

With the fast development of high speed wireless technologies and the growing population of mobile portable devices, location based information is potentially available for access control systems including rich context information about both the users and the resources they access. Location-based service provisioning is of great interest to wireless Internet service providers to deliver attractive value-added services. Specifically designed location based applications are especially useful in emergency situations, where quick responses are urgently required for persons to be physically present in a certain place to perform sensitive tasks without conflicting with security policies. For example, in the aviation industry, highly complex and delicate processes have to be executed on a routine basis in order to guarantee correct maintenance of airplanes before a flight. Maintenance technicians check the important capabilities of the planes, often requiring a second check by another technician. Also the technicians need to be agile while inspecting, such as moving in and around the airplane. This makes the use of handheld devices appealing, which allows the technicians to commit certain tasks of the process in specific locations, e.g., in a safe position or the only position where they can effectively perform the required actions. Furthermore, when exceptional vulnerabilities of parts have been detected, the maintenance process needs to be quickly adapted without violating security constraints of the original inspection process. Also in cases of emergency, personnel needs to be warned or called to the scene for assistance. This example illustrates the following distinct security and business requirements for mobile collaboration:

Flexible business processes: Business processes may require changes anytime anywhere due to urgent requirements, such as the maintenance process. To cope with runtime changes, information systems that manage the business processes have to be flexible and efficient in adapting the existing business processes.

Secure authorizations: Since execution of a business process may often involve multiple people in different roles to perform activities handling sensitive information, secure authorization is very important. Generally, security requirements are specified as security constraints, such as Separation of Duty (SoD) constraints that restrict the access rights to sensitive tasks in order to prevent users from exceeding

978-0-7695-3708-5/09 $25.00 © 2009 IEEE
DOI 10.1109/SERVICES-I.2009.79

their authority. For instance, the routine double check of an airplane should be accomplished by two engineers.

Qualification and context awareness: To ensure the effect of human-involved activities, certain qualifications are required of the performers. Practically, there often exist a few candidate performers qualified for an activity. It is desired to select the most *suitable* user taking the location context of the performers and the scene into account to fulfill the necessary business process as soon as possible.

Simultaneously satisfying the above goals is challenging since it is difficult achieve flexibility without compromising security and manageability. The change of business processes and the required performers are unknown before an emergency and thus the system administrator cannot adequatly arrange every performer of tasks in advance. Therefore quick response systems should provide a collaborative capability in effectively scheduling the involved activities, while enforcing the corresponding security constraints, particularly considering the location context as well as the performers' identity and attributes.

Previous literature has discussed some aspects of above considerations without a holistic solution. The context-aware access control models and systems, such as Task-Based Authorization Control model (TBAC) [18], Activity-based Access Control model [19], Task-role-based Control Model (T-RBAC) [13] and the logic based workflow system (W-RBAC)[20], etc., can specify security constraints on business processes and enable permission granting, tracking, and revoking to be automated and coordinated with the processing of tasks. Location based information has been used in different access control models to allow taking a users' physical location into account when determining their access privileges [3], such as the extension of the Mandatory Access Control (MAC) model [15], the extension of the widely adopted RBAC model [16], and GEO-RBAC [5]. However, none of these models take all the considerations together in an access control system, especially they do not tackle the problem of finding the users most suitable for a quick response.

In this paper, we investigate this challenging problem and propose the Constraint-based Authorization Management Model for Mobile Collaboration Services (CAM). The notion of *Location Based Authorization Execution Binding* is introduced in CAM to activate authorizations in a certain business instance in consideration of the users' location, identity, and attributes. It is helpful for choosing the most *suitable* user fulfilling the quick response requirements while satisfying the specified security constraints of the dynamically changing business processes. We also discuss the Quick-Response Authorization Problem (QAP) for mobile collaboration and analyze its computational complexity. The algorithm of execution planning for a dynamic business process is presented to ensure that a process does not block due to security constraints. Finally we propose a system architecture based on a Web service implementation of XACML [12] and a BPEL engine with the BPEL4People [1] extension.

The paper is organized as follows. The following section introduces the main concepts of the proposed access control model CAM. After that, Section 3 discusses the Quick-Response authorization problem for mobile collaboration. Section 4 introduces the proposed architecture based on Web services, BPEL, and XACML with an example of a possible policy syntax using extended XACML policies. Finally Sections 5 and 6 expand on related work, conclude the paper and show future research directions.

2 The Constraint Based Authorization Management Model

The core of our approach is the proposed Constraint-based Authorization Management Model for Mobile Collaboration (CAM). This model is an extension to RBAC, motivated by the fact that Role Based Access Control (RBAC) is today's most recognized access control model [17]. In RBAC, users gain permissions through memberships in roles. There are a number of constraint mechanisms to express security requirements, such as mutual exclusion. These are widely used to support the properties of Separation of Duty. Our proposed model is shown in Figure 1. The solid lines signify assignments during the modeling phase, the dashed lines describe assignments that are dynamically made at runtime. The definition of each component is given below.

2.1 Basic Terminologies in CAM

Users. Users are the subjects of access control and are assigned responsibilities to perform certain job functions. In a business process, a user is assigned to one or more roles that can enable the user to perform activities to complete a part of the business process. The specification of a user u is of form $< id, attrs, state >$, where id is the identifier of u, $attrs$ are the attributes u holds and $state \in \{on\_duty, free\}$ denotes the state whether u is available. The set of users in a given system is denoted as U.

Roles. Roles are authorized to perform an activity of a business process and are hierarchically related. The specification of a role r includes the qualification requirement on the attributes of its members, denoted as $r.qua$, which may take the form of an expression consisting of user-attributes and operators in $\{\neg, \vee, \wedge\}$. A term of $r.qua$ can be of the form $(a.i \text{ op } c)$ or $a.attrs \in C$, where $a.i$ is one of the user attributes, $op \in \{=, \neq, <, >, \leq, \geq\}$, c is a condition on user attributes, and C is a set of such conditions. For example, $(Location =$ "New York City"$) \wedge \neg(Position \in$

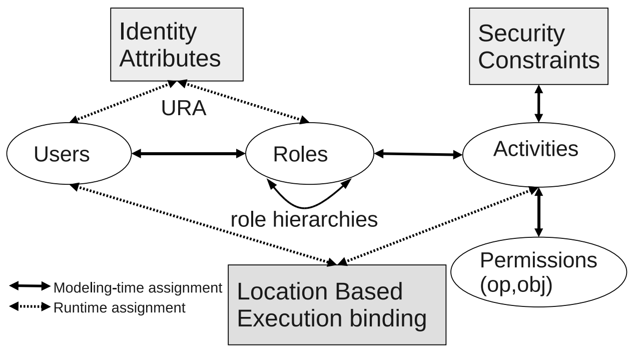

Figure 1. The Constraint-based Authorization Management for Mobile Collaboration.

{"Manager", "Director"}). Role hierarchies $RH \subseteq R \times R$ are partial orders on a role set R and define inheritance relations among roles, written as $\succeq$. The expression $r_i \succeq r_j$ means that users who are members of r_i are also members of r_j, while all permissions assigned to r_j are inherited by r_i.

Attributes Based User Role Assignments. The User Role Assignments $UR \subseteq U \times R$ are implemented by evaluating a users' attributes against a roles' qualification requirements. A role r of user u is activated when her attributes satisfy the role qualification, formally

$Sat(u.attrs, r.qua) \models (u, r) \in UR$

Let $SatU(r : R) \rightarrow 2^U$ denote the set of users qualified for r. Then, $SatU(r) = \{u|u \in U \wedge Sat(u.attrs, r.qua)\}$.

However, a user being assigned to a role does not mean she can definitely execute that role to perform an activity in a business process instance since the assignments should also satisfy the specified security constraints and depend on the users' location. The activation of authorizations is implemented in the part of CAM *location based runtime binding*, discussed in Section 3.1.

Permissions and activities. Permissions are the approvals of users to operate on one or more protected objects in a step of a business process. Permissions are associated with activities in our model rather than with roles in the RBAC model, which can restrict permissions to be active only during execution of the activity. This ensures that activities are executed by the right person and at the right time. An activity defines an atomic task of a business process that is distinct from other tasks and executed separately. The specification of an activity a is of the form $\langle id, p\_set, r\_set\rangle$, where id is the identifier, $p\_set \subseteq P$ is the set of permissions encapsulated in atv, and $r\_set \subseteq R$ is the set of roles which are authorized to perform a. Activities can be mapped to human tasks in BPEL processes.

Let P and A respectively represent the set of permissions and activities in a system. We define two predicates $ActU(u : U) \rightarrow 2^A$ to calculate the activities user u is allowed to perform, and $UserA(a : A) \rightarrow 2^U$ to calculate all the users who are authorized to perform activity a, respectively:

$ActU(u) = \{a|a \in A \wedge (r \in a.r\_set \vee r \in \{r|r' \in a.r\_set \wedge r \succeq r'\}) \wedge Sat(u.attrs, r.qua)\}$

$UserA(a) = \{u|u \in U \wedge Sat(u.attrs, r.qua) \wedge r \in a.r\_set \vee (r \succeq r' \wedge r' \in a.r\_set)\}$

Security Constraints and Consistency. Constraints are a fundamental mechanism in RBAC systems to enforce high level security requirements. The most frequently considered constraints in business process environments are Mutual Exclusion (ME) and Binding of Duty (BD), which have been well established in previous literature [14]. The semantics of each constraint are given below. We would like to state that all constraints in this model are specified on activities since all fundamental performance in business processes is based on activities.

Mutual Exclusion ME(A_s): This constraint requires that no user is allowed to be authorized to more than one activity in the set $A_s \subseteq A$. For example, ME$(\{IssueCheck, InitCheck\})$ enforces that a user is only allowed to either execute the activity $initiate\ a\ check$ or $issue\ the\ check$. This most frequently mentioned constraint is a powerful means of limiting the distribution of critical permissions and is widely used to support the Separation of Duty policy or to enforce the conflict of interest policy. This constraint can be further classified into static mutual exclusion and dynamic mutual exclusion according to whether it is enforced on model or instance, denoted as SME and DME respectively. The static Mutual exclusion SME sets up constraints of the user-role-activity assignments such that no user is allowed to take on more than one activity in A_s, while the Dynamic Mutual exclusion DME sets up constraints on the enabled authorization for each business instance during execution such that a user is allowed to execute more than one activity in A_s but can be enabled at most once per process instance.

Binding of Duty BD(A_s): The constraint restricts the set of activities $A_s \subseteq A$ that may be performed by the same user. The purpose of the binding constraint is to simplify the management of users and roles by requiring that if a user takes a certain responsibility, then she must also take another responsibility. For example, BD$(\{Init_{emerg}, Report_{emerg}\})$ requires that any user who initiates a emergency process must also later execute create a report for this emergency.

There often coexist multiple constraints in a system that interact with each other and with the role hierarchies. However, multiple constraints together may preclude one from assigning any user to a task. For example, if a ME constraint requires that no user is authorized for both activities a_1 and a_2, yet a_1 and a_2 are also associated with a BD constraint (have to be performed by the same user), it is therefore impossible to assign these activities to users without violating constraints. Hence, consistency checking is desired to identify and resolve conflicts in existing constraints. Obviously, for any pair of constraints $\mathsf{ME}(A_s)$ and $\mathsf{BD}(A'_s)$, $|A_s \cap A'_s| \leq 1$ should hold.

Overall, all the above entities and relationships together present an environment for a business process to be run in, which is called the *configuration* of a system.

Definition 1 (Configuration) A *configuration* is given as a tuple $\langle U, R, RH, P, A, C\rangle$, where U is a set of users, R is a set of roles, $RH \subseteq R \times R$ defines role hierarchies, P defines a set of permissions, A is the set of activities, and C is a set of security constraints that should be followed in any process instance and each constraint is of the form $\mathsf{ME}(A_s)$ or $\mathsf{BD}(A'_s)$, where $A_s, A'_s \subseteq A$.

3 Authorization Activation for Mobile Collaboration

Generally, a business process instance (*instance* for short) has the form $\langle BP, \preceq\rangle$, where $BP \subseteq A$ is a subset of activities and $\preceq \subseteq BP \times BP$ defines a partial order among activities in a BP. In this section, we will discuss how to identify the appropriate user role assignment for performing activities in an instance.

3.1 Location Based Authorization Execution Binding

A user being assigned to a role to perform activities only means that she is competent for those activities, but it does not mean that she will be involved in every instance. The authorization activation in an instance also depends on three other conditions: 1) whether the assignment satisfies all security constraints; 2) the user is available at that moment, i.e., her status is $free$; and 3) she is within a *reasonable* distance regarding the scene. Such activation of authorization is called the *Location Based Authorization Execution Binding* in our model. Since in emergency situations quick response is urgently required for activity performers to be present at a scene, the closest qualified users who also satisfy the security constraints are most suitable to be involved in an instance. Therefore, we define three classes of location-based predicates to assess a certain user's position, given as follows.

User-specific predicates $Dis(u, pos)$ assess the distance of a specific user u to a given position pos. It is generally obtained from a given mobile terminal or a cellular phone.

Area-specific predicates $InArea(region)$ identify all the users in a specific area $region$, either via a geometric model (i.e., a range in a n-dimensional coordinate space) or a symbolic model (i.e., with reference to entities of the real world such as cells, streets, cities, zip code, buildings, etc.) [10].

User-area determining predicates $Deter(u, region)$ determine whether user u is in a specific area $region$. For instance, to evaluate whether a user is in a certain building or city or in the proximity of other entities.

3.2 The Quick-Response Authorization Problem

Definition 2 (Quick-Response Assignment) Given a configuration $\langle U, R, RH, P, A, C\rangle$, a business process $\langle BP, \preceq\rangle$, a position pos and area $region$, $URA \subseteq U \times A$ is called an *Quick-Response user-activity enabled assignment* (*QR assignment*) if and only if the following four conditions are true: 1) Every activity $a \in BP$ is assigned to one user; 2) For each $(u, a) \in URA$, there exists a role r such that $Sat(u.attrs, r.qua) \wedge r \in a.r\_set$; 3) No constraint in C is violated and 4) The distances of the users in URA to pos is closer than other assignments. Formally, $min\{max_{u \in Inarea(region) \wedge (u,a) \in URA \wedge a \in BP} dist(u, pos)\}$.

Note that in Definition 2, we require that every role must be assigned to one user so that the tasks represented by them can be performed. The distance requirement is based on the assumption that the response time for any performer to be present at the scene only relates with the distance without consideration of other factors, such as the traffic. Additionally, there may exist multiple assignment plans of authorization. For example, there are two activities a_1 and a_2 in an instance, and a constraint $\mathsf{ME}\{a_1, a_2\}$. Users u_1 and u_2 are qualified for a_1, while u_3 and u_4 are qualified for a_2. Then, the candidate authorization plans are: $\{u_1, u_3\}$, $\{u_2, u_3\}$, $\{u_1, u_4\}$, and $\{u_2, u_4\}$ for both activities. If the evaluated results of user positions are: $Dis(u_1, pos) < Dis(u_2, pos) < Dis(u_3, pos) < Dis(u_4, pos)$, then we would choose the assignment $\{u_1, u_3\}$ since the response time has the upper bound of the farthest one in a plan. Thus, a meaningful problem is looking for the *QR assignment* for an instance $\langle BP, \preceq\rangle$ under a configuration $\langle U, R, RH, P, A, C\rangle$, which is called the *Quick-Response Assignment Problem (*QAP*)* in this paper.

We will show that QAP in the most general case (i.e., with all types of constraints) is **NP**-hard. In order to understand how different types of constraints affect the computational complexity of QAP, we will discuss QAP with consideration of different types of constraints in C. For example, $\mathsf{QAP}\langle \mathsf{BD}, S\mathsf{ME}\rangle$ represents the subcase where Binding of Duty and Static Mutual Exclusion constraints are the only two types of constraints that may be used, while $\mathsf{QAP}\langle \mathsf{BD}, \mathsf{ME}\rangle$ is the general case.

Theorem 1 $\mathsf{QAP}\langle \mathsf{BD}, S\mathsf{ME}\rangle$ *is in P.*

Proof 1 *A* $S\mathsf{ME}\{A_s\}$ *constraint means that no user is allowed to take more than one activity in* A_s. *Formally,* $\forall a_1, a_2 \in A_s, UserA(a_1) \bigcap UserA(a_2) = \emptyset$,

```
Initialization: $DIS_{max} = 0$; $EnU = \emptyset$; $DIS_{opt} = \infty$; $URA_{opt} = \emptyset$
1. FOR EACH activity $a$ in the business process $\langle BP, \preceq \rangle$
        $Lock(a) = 0$;
        calculate the predicates $UserA_{region}(a)$ under $\langle U, R, RH, P, A, C \rangle$ and $region$
        $UserA_{region}(a) = \{u | deter(u, region) \wedge Sat(u.attrs, r.qua) \wedge r \in a.r\_set \vee (r \succeq r' \wedge r' \in a.r\_set)\}$
        IF $UserA_{region}(a) = \emptyset$ return FALSE
2. FOR EACH BD constraint $c \in C$ DO
        IF $\bigcap_{a \in c.A_s} UserA_{region}(a) = \emptyset$ THEN return FALSE;
        ELSE FOR each $a \in c.A_s$ DO $UserA_{region}(a) = \bigcap_{a \in c.A_s} UserA_{region}(a)$;
3. FOR EACH activity $a \in BP$ DO
        IF $|UserA_{region}(a)| = 1$ THEN $Lock(a) = 1$; $EnU = EnU \cup UserA_{region}(a)$;
4. Generate an enable assignment $URA$ for the activities with $a \in BP \wedge Lock(a) = 0$
        IF $URA$ satisfies the related ME constants
                THEN FOR each $(u, r) \in URA$ DO
                        $EnU = EnU \cup \{u\}$
                        IF $dist(u, pos) > DIS_{max}$ THEN $DIS_{max} = dist(u, pos)$;
                IF $DIS_{max} < DIS_{opt}$ THEN $DIS_{opt} = DIS_{max}$; $URA_{opt} = URA$;
5. Return $URA_{opt}$.
```

Figure 2. The Execution Planning for Business Process.

where $UserA(a_i)$ are the users who are authorized to perform the activity a_i, as defined in Section 2.1. Now, we construct the QR *assignment URA for $\langle BP, \preceq \rangle$ under $\langle U, R, RH, P, A, C \rangle$ in the three steps:*

First, for each activity $a \in BP$, we calculate all candidate users in the given region, namely $UserA_{region}(a) = \{u | deter(u, region) \wedge Sat(u.attrs, r.qua) \wedge r \in a.r\_set \vee (r \succeq r' \wedge r' \in a.r\_set)\}$, which obviously can be done in polynomial time. Secondly, to ensure that the BD *constraints are always satisfied, we execute some pre-processing before choosing users for activities. For each* BD$(A_s) \in C$*, we set the set of users that are authorized to perform these activities A_s to the intersection of the set of users who are authorized to perform each activity. Formally, $\forall a_i \in A_s, UserA_{region}(a_i) = \cap_{a_i \in A_s} UserA_{region}(a_i)$. Furthermore, for each S*ME(A_s)*, no user is allowed to take more than one activity in A_s. Thus no eligible user for an activity $a \in BP$ violates the S*ME *constraints. Thirdly, for each activity $a \in BP$, we select the nearest user from the candidates, namely the user with $min\{dist(u, pos) | u \in UserA_{region}(a)\}$.*

From above construction, we can see that the generation of assignment URA could be done in polynomial time with the upper bound of $O(|BP| * |U| * |R|)$ and URA satisfies the four conditions in definition 2.

Theorem 2 QAP$\langle$ME, BD$\rangle$ *is* **NP***-hard.*

We prove this theorem by reducing the **NP**-complete Graph K-Color Problem to QAP. The general case of QAP is intractable which means that there exist instances that take exponential time in the worst case. However, many instances that will be encountered in practice may be efficiently solvable. Our ultimate goal is to find a *QR assignment* for a given $\langle BP, \preceq \rangle$ under configuration $\langle U, R, RH, P, A, C \rangle$. Practically, not every configuration has a valid assignment that can be enabled, let alone a quick response choice. For example, if Alice is the only user qualified for activities a_1 and a_2 and there is a constraint ME$(\{a_1, a_2\})$, then there is no way to assign both activities to a user without violating the mutual exclusion constraint. So, it is necessary to check whether there is an assignment that could be enabled, furthermore if there is more than one choice the question arises, which one can respond most quickly. We will discuss this process in the next section.

3.3 Instance Execution Planning

In this section, we will present an execution planning algorithm to find the *QR assignment* for a given configuration $\langle U, R, RH, P, A, C \rangle$ and an instance $\langle BP, \preceq \rangle$. The algorithm is shown in Figure 2. We first enumerate all qualified users $UserA_{region}(a)$ in the certain area $region$ for each activity by dint of the location based services $deter(u, region)$. If the result is empty then there is no qualified user to perform this activity and therefore no feasible authorization solution for the business process. In step 2, we perform a heuristic optimization to minimize the complexity by setting the candidate users for each activity involved in a BD constraint to the same set of users. For example, if there is a binding of duty constraint BD$(\{a_1, a_2\})$ and the qualified user for each activity are $\{u_1, u_2, u_3\}$ and $\{u_4, u_2, u_3\}$, respectively. Then we set the intersection set $\{u_2, u_3\}$ as their candidate users since the binding constraint requires activities a_1 and a_2 to be always performed by the same user. Such manipulation ensures that the BD constraints is always satisfied.

Then we fix the performers to those activities in the case that they are the only qualified users in step 3. This heuristic optimization can greatly reduce the attempts of impossible authorizations. For example, if there is ME$(\{a_1, a_2\})$ constraint associated with activities a_1 and a_2 and there are two users u_1 and u_2 eligible for a_1 while only one user u_1 is eligible for a_2. In this case, if a_1 is performed by u_1, a_2 cannot be executed by u_1 again, which halts the execution of the instance. Therefore, we lock a_1 with u_2 and a_2 with u_1 to guarantee the successful execution of the whole business process.

Finally, we enumerate all possible authorization plans for the instance under $\langle U, R, RH, P, A, C\rangle$ and check which one can respond most quickly by verifying the location of the performers. This step also guarantees that all ME constraints are satisfied and the selected users are in the $quick\ response$ group in terms of the distance. Although the complexity of this step is $O(|BP|^{max\{UserA_{region}(a)\}})$, in practice the adoption of above two heuristic optimizations can greatly reduce the complexity of computing the possible attempts and increase the rate of success.

4 System Architecture

In this section, we discuss how the above requirements and algorithms can be integrated into a Workflow Management System based on Web service technologies. We are currently in the phase of prototypically implementing this architecture.

Modern open-source BPEL engines like ActiveBPEL [2] already support human tasks as described in the BPEL4People specification [1]. We propose to extend a BPEL4People engine with the access control architecture for human tasks as shown in Figure 3. Once a user logs into the Portal, she is presented a task list according to her role. When the user is available to execute a task the portal makes a SOAP request to the Policy Enforcement Point (PEP) (step 1). The PEP has the responsibility to evaluate which tasks the user can execute based on the user's role, context variables, like user location, and complex constraints like Separation of Duty. The PEP consults the Policy Decision Point (PDP) whether the user has the permission to execute the given activity (step 2). The Web service interface could be implemented with the Web service based XACML PDP implementation XACMLLight [6]. The PDP then fetches the applicable XACML policies from the policy repository and retrieves required context variables, like the user's location, from the context provider (steps 3-4). Depending on the requirements on integrity and robustness, the location information can simply be sent via the access request from the mobile device and additionally double checked by the service provider. If the PDP decides that, according to the policies, the user is allowed to execute the task, the Execution Planning Engine is invoked to check whether the execution is in conflict with higher level security constraints. This engine has access to the policy repository and the current user role assignments to evaluate whether the execution of the activity will cause a deadlock in the business process

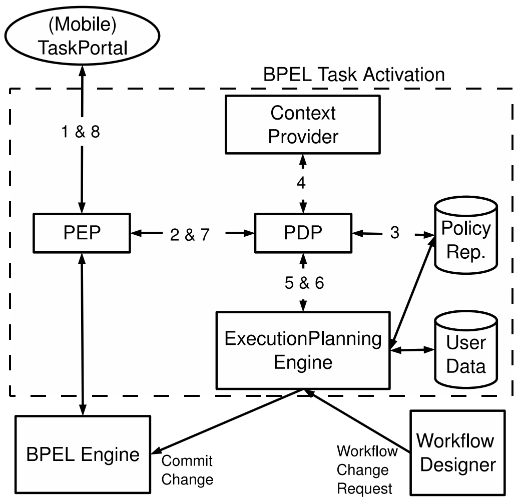

Figure 3. The proposed system architecture based on BPEL4People and XACML

due to security constraints (steps 5-6). If this evaluation also returns no restriction on the execution of the task the PDP returns "PERMIT" to the PEP allowing the user to execute the task. Furthermore, if a SoD constraints was set for the given activity the PEP has to fulfill the obligation of writing a new policy to the policy repository that enforces the constraint for that user (as suggested in [4]). Of course, a sophisticated rollback mechanism has to be implemented in case the user does not execute the task.

To enable the constraint validation for a flexible business process, the execution planning engine should be consulted prior to each change that a process designer wants to conduct on a running process instance. If an inconsistency is found, the change of the process should be blocked. This architecture can be particularly useful for implementors since current BPEL engines only have limited runtime adaption capabilities [8].

4.1 Policy Specification in XACML

In this section we present how the required policies can be encoded into XACML policies. The following example refers to a business process aided airplane inspection where the technicians are equipped with mobile devices. The policy shown in Listing 1 allows a user with the role technician to execute the activity *checkSystem* (lines 2-19) if she is located in the location with the semantic name *cockpit* (lines 20-29) and has not executed the same task before (lines 30-37). Furthermore, if access is granted, the PEP has the obligation to store the event of the task execution to the history. Also in the example the *checkSystem* task has to be executed twice by two different technicians. This is realized with the extend XACML function *hasExecutedActivityBefore* which looks up in the execution history whether the user has executed the task before in the same business process instance. Obviously an XACML based architecture has to implement the functions *semantic-location-match*, *hasExecutedActivityBefore* and needs to provide the location sensing functionalities, that in this context could be realized via sensors in the airplane cockpit.

```
<Policy PolicyId="SoDandLocationPolicy" RuleCombiningAlgId="...deny-overrides">
 <Target>
  <Subjects>
   <Subject>
    <SubjectMatch MatchId="...string-equal">
     <AttributeValue DataType="...string">technician</AttributeValue>
      <SubjectAttributeDesignator AttributeId="...role" DataType="...string"/>
    </SubjectMatch>
   </Subject>
  </Subjects>
  <Actions>
   <Action>
    <ActionMatch MatchId="...string-equal">
     <AttributeValue DataType="...string">checkSystem</AttributeValue>
     <ActionAttributeDesignator AttributeId="..activity" DataType="...string"/>
    </ActionMatch>
   </Action>
  </Actions>
 </Target>
 <Rule Effect="Permit" RuleId="locationRule">
  <Target />
   <Condition>
    <Apply FunctionId="semantic-location-match">
     <AttributeValue DataType="...string">cockpit</AttributeValue>
      <SubjectAttributeDesignator AttributeId="user-location"
           DataType="...string" Issuer="locationVerificaitonService"/>
    </Apply>
   </Condition>
 </Rule>
 <Rule Effect="Deny" RuleId="SoDRule">
  <Condition>
   <Apply FunctionId="hasExecutedActivityBefore">
    <SubjectAttributeDesignator AttributeId="userID" DataType="...string"/>
    <AttributeValue DataType="...string">checkSystem</AttributeValue>
   </Apply>
  </Condition>
 </Rule>
 <Obligations>
  <Obligation FulfillOn="Permit" ObligationId="safeExecutionToHistory" />
 </Obligations>
</Policy>
```

Listing 1. XACML Syntax for CAM **policies**

5 Related Work

Location-based Access Control (LBAC) techniques allow taking a users physical location into account when determining their access privileges. Ardagna et al. [3] integrate location-based conditions along with a generic access control model to grant or deny access by checking the requester's location as well as credentials. The extension of Mandatory Access Control (MAC) in [15], also incorporates the notion of location and uses the location information to determine whether a subject has access to a given object. Their work only considers integrating the location based information into the access control decision, such as the specification and enforcement of location based conditions, while our work focuses on the authorization activation suitable for mobile collaboration where the activities may be dynamically changed at runtime and the quick response authorizations are required.

Some literature tries to integrate the location based information into RBAC [17] to satisfy certain geographic requirements [16]. The most sophisticated location based access control model is GEO-RBAC [5]. It uses standard geospatial information to gain semantic knowledge of geospatial features and to map between coordinates and the logic location of users and objects. Their work discusses how each component of RBAC, such as objects, user positions, geographically bounded roles, and security constraints, are supported with spatial and location based information, which provide a comprehensive framework to deal with spatial aspects in real mobile applications. However, in urgent situations, quick enforcement of authorization is desired and the performers are required to be present at the scene, which was not mentioned in their work.

Our work is also related with the field of context-aware access control and the specification of security constraint on business processes, such as the Task-Based Authorization Control model (TBAC) [18], the Activity-based Access Control model [19], and the Task-role-based Control Model (T-RBAC) [13]. However, from a conceptual standpoint, our considerations and method are significantly different and comprehensive. These works mostly support the active authorization for a predefined workflow, while we focus on secure authorizations in an active collaboration where the component activities may be changed at runtime. Wainer et al. [20] present a logic based workflow system (W-RBAC) that allows the specification of organizational hierarchies and Separation of Duty constraints among others. Compared with our model, W-RBAC does not propose an algorithm for execution planning. Weber et al. [21] also discuss access control in a business process context but they only focuses on the rights to manage, create, and adapt workflows themselves. In contrary to our model, Weber et al. neither cover execution rights nor do they tackle complex constraints like Separation of Duty. Koufi et al. [9] take users location into account for a BPEL-based workflow system, but do not cater for flexible business process or provide a formal model. Mendling et al.[11] describe how Separation of Duty constraints can be expressed for BPEL4People workflows. Hong et al. [7] considers various computing context, user context, and physical context, and propose a conceptual model and methodology for adapting existing enterprise services into ubiquitous ones, but without the consideration of security. Overall, to the best of our knowledge, there is no previous work providing a holistic solution of location based security mechanisms for changing mobile business processes.

6 Conclusions and Future Work

Under emergency situations, quick response is urgently required for persons to be physical present in a certain place to perform sensitive tasks without compromising security policies, where the required business process may be dynamically changed and the exact performers are uncertain. To tackle this challenging problem, we propose the Constraint-based Authorization Management (CAM) model, in which the flexibility of business process is implemented by adaptively assembling activities at runtime, while the secure authorizations are ensured by specifying security constraints on activities and by verifying its consistency at both design- and executiontime. To allow for the identification of users that are physically closest to the position needed to execute a task, the notion of Location Based Authorization Execution Binding and an execution planning algorithm are introduced with the help of the location based predicates. We have also proposed a BPEL4People and XACML-based implementation of the CAM model as well as a possible syntax to specify access control policies in XACML.

Although our model and methodologies present a pioneer work for the quick response problem in the securely flexible collaboration, there still exist many more interesting problems. As part of future work, we are planning to consider more sophisticated conditions to make more precise decisions. In the current study, we do not consider the duration of each activity as a limiting factor. In practice, each activity in a business process may last a period of time and the sequence between them influences the total response time. Other context attributes [7], such as traffic conditions, weather, etc., determine the elapsed time for a person to be present at a scene. Hence, it would be interesting to take these considerations also into account when solving QAP. Another future research direction is to investigate the secure authorization delegation problem in the flexible business process to handle exceptions and alternatives. We are also implementing the proposed system presented in this paper.

References

[1] Active Endpoints Inc. et al. Ws-bpel extension for people (bpel4people), version 1.0. June 2007.

[2] ActiveVOS. The activebpel community edition engine. http://www.activevos.com/community-open-source.php (4/22/09), 2009.

[3] C. A. Ardagna, M. Cremonini, E. Damiani, S. D. C. di Vimercati, and P. Samarati. Supporting location-based conditions in access control policies. In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security table of contents*, pages 212–222, New York, NY, USA, 2006. ACM.

[4] J. Crampton. Specifying and enforcing constraints in role-based access control. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*, pages 43–50, New York, NY, USA, 2003. ACM.

[5] M. L. Damiani, E. Bertino, B. Catania, and P. Perlasca. GEO-RBAC: A spatially aware RBAC. *ACM Trans. Inf. Syst. Secur.*, 10(1), 2007.

[6] O. Gryb. XACMLLight. http://sourceforge.net/projects/xacmllight (4/22/09), 2009.

[7] D. Hong, D. Chiu, S. Cheung, V. Shen, and E. Kafeza. Ubiquitous enterprise service adaptations based on contextual user behavior. volume 9, pages 36–44. Springer, 2007.

[8] D. Karastoyanova, A. Houspanossian, M. Cilia, F. Leymann, and A. Buchmann. Extending bpel for run time adaptability. *Enterprise Distributed Object Computing Conference, IEEE International*, 0:15–26, 2005.

[9] V. Koufi and G. Vassilacpoulos. Context-aware access control for pervasive access to process-based healthcare systems. *Studies in health technology and informatics (Stud Health Technol Inform)*, 136:679–684, 2008.

[10] N. Marsit, A. Hameurlain, Z. Mammeri, and F. Morvan. Query processing in mobile environments: a survey and open problems. In *In Proc. of the First International Conference on Distributed Framework for Multimedia Applications (DFMA-05)*, pages 150–157, Washington, DC, USA, 2005. IEEE Computer Society.

[11] J. Mendling, K. Ploesser, and M. Strembeck. Specifying separation of duty constraints in bpel4people processes. In W. Abramowicz and D. Fensel, editors, *Proceedings of the 11th Int'l Conference on Business Information Systems (BIS 2008)*, LNBIP, pages 273–284, Innsbruck, Austria, 2008. Springer Verlag.

[12] T. Moses (Editor). eXtensible Access Control Markup Language (XACML) Version 2.03. Technical report, February 2005.

[13] S. Oh and S. Park. Task-role-based access control model. *Inf. Syst.*, 28(6):533–562, 2003.

[14] F. Paci, R. Ferrini, Y. Sun, and E. Bertino. Authorization and user failure resiliency for ws-bpel business processes. pages 116–131. Springer, 2008.

[15] I. Ray and M. Kumar. Towards a location-based mandatory access control model. *Computers and Security*, 25(1):343–358, 2006.

[16] I. Ray and M. Toahchoodee. A spatio-temporal role-based access control model. In *Proceedings of the 21th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, pages 211–226, Berlin / Heidelberg, Germany, 2007. Springer.

[17] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[18] R. K. Thomas and R. S. Sandhu. Task-based authorization controls (tbac): A family of models for active and enterprise-oriented autorization management. In *Proceedings of the IFIP TC11 WG11.3 Eleventh International Conference on Database Securty XI*, pages 166–181, London, UK, UK, 1998. Chapman & Hall, Ltd.

[19] W. Tolone, G.-J. Ahn, T. Pai, and S.-P. Hong. Access control in collaborative systems. *ACM Comput. Surv.*, 37(1):29–41, 2005.

[20] J. Wainer and P. Barthelmess. W-rbac - a workflow security model incorporating controlled overriding of constraints. *International Journal of Cooperative Information Systems*, 12, 2003.

[21] B. Weber, M. Reichert, W. Wild, and S. Rinderle. Balancing flexibility and security in adaptive process management systems. In *OTM Conferences (1)*, pages 59–76, 2005.