2008 IEEE Workshop on Policies for Distributed Systems and Networks

Consent-based Workflows for Healthcare Management

Giovanni Russello, Changyu Dong, Naranker Dulay
Department of Computing
Imperial College London
180 Queen's Gate, London, SW7 2AZ, UK
{g.russello, changyu.dong, n.dulay}@imperial.ac.uk

Abstract

In this paper, we describe a new framework for healthcare systems where patients are able to control the disclosure of their medical data. In our framework, the patient's consent *has a pivotal role in granting or removing access rights to subjects accessing patient's medical data. Depending on the context in which the access is being executed, different consent policies can be applied. Context is expressed in terms of workflows. The execution of a task in a given workflow carries the necessary information to infer whether the consent can be* implicitly *retrieved or should be* explicitly *requested from a patient. However, patients are always able to enforce their own decisions and withdraw consent if necessary. Additionally, the use of workflows enables us to apply the* need-to-know *principle. Even when the patient's consent is obtained, a subject should access medical data only if it is required by the actual situation. For example, if the subject is assigned to the execution of a medical diagnosis workflow requiring access to the patient's medical record. We also provide a complex medical case study to highlight the design principles behind our framework. Finally, the implementation of the framework is outlined.*

1 Introduction

Computerisation of healthcare information storage and processing, while offering new opportunities to improve and streamline healthcare delivery, also presents new challenges to individual privacy. Healthcare information refers to "*any information, whether oral or recorded in any form or medium, that: (1) Is created or received by a health care provider, health plan, public health authority, employer, life insurer, school or university, or health care clearinghouse; and (2) Relates to the past, present, or future physical or mental health or condition of an individual; the provision of health care to an individual; or the past, present, or future payment for the provision of health care to an individual [2].*" Healthcare information contains sensitive personal information; i.e. it may include the details of a person's history of diseases and treatments, history of drug use, genetic testing, sexual orientation and practices etc. Improper disclosure of this data can influence decisions about an individual's access to credit, education and employment. Therefore, it is crucial that healthcare information systems, which allow electronic storage, transmission, display and analysis of healthcare information, should offer adequate protections to address these concerns.

It has been well accepted in modern medical ethics and law that *patient information is confidential and should not be disclosed without adequate justification. The justification for disclosure should normally be consent* [1]. However, most security models for clinical information systems are merely variations of Role-Based Access Control (RBAC) which make access decisions based on the role of the user rather than patient consent. There are some exceptions, for example the BMA policy model [5, 6] and Cassandra [10, 11]. The BMA policy model is the first security model which requires the patient's consent for accessing healthcare information. Cassandra is a trust management system designed for securing electronic health records which captures consents as special roles in the system. Nevertheless, they have some common problems. First, how to capture patient consent properly. Patient consent can be *explicit*, e.g. in written form, but more often is *implicit*, e.g. the context in which the access is being executed could carry enough information for implicitly obtain consent. In general, when the use and disclosure of patient information is for the patient's own healthcare purposes, and provide the patient or his legal representative has been informed of what information sharing is necessary for such purposes, implicit consent is sufficient. But in the BMA model and Cassandra, the consent must be explicit. This requirement adds unnecessary workload to healthcare professionals. Second, how to ensure that the consent is obtained on a well informed basis. A valid consent requires that the patient has been

978-0-7695-3133-5/08 $25.00 © 2008 IEEE
DOI 10.1109/POLICY.2008.22

informed as to what information is intended to be used or disclosed, and for which purposes. Consent that has been obtained does not imply information has been given. However, none of the current models handle this.

In this paper, we address two main problems of current access control mechanisms. The first problem is the integration of patient consent with access control. Most access control models are designed for non-healthcare systems and do not have the concept of patient consent at all. However, patient consent occupies a pivotal role in legitimising the use and disclosure of healthcare information. Patients have a right to control access to and disclosure of their own healthcare information by giving, withholding or withdrawing consent. Therefore patient consent should serve as the ultimate foundation of access control decisions in healthcare systems. The second problem is related to the **need-to-know** principle. According to this principle, even when one has the necessary approvals, i.e. if the patient consents, access should not be given unless one has a specific need to know. An access decision should be justified by not only who is requiring access and what is being accessed, but also why the information needs to be accessed. Capturing and enforcing the access is also useful for mitigating exposure of healthcare systems to insider attacks [4, 5]. For example, browsing a patient's medical record by a doctor should be allowed when the doctor is diagnosing the patient, but should not be allowed if the doctor is off-duty. Access control models such as RBAC cannot capture the access needs precisely. For example, in RBAC, the permission assignment is decided by "job functions" assigned to the role. The set of permissions is assigned to the role statically to enable the subjects playing this role to perform all the job functions. The subject has all the permissions all the time even when they are not performing certain job functions.

We believe that to better protect patient privacy, access control should not just be based on rules of who should or should not access what. Enforcing a correct procedure is also important. That is why we introduce a workflow based control framework. The framework is designed for healthcare systems and can enforce consent-based access control as well as the need-to-know principle and various other constraints. In addition, it releases end users (e.g. the medical professionals) from security related configurations so that they can concentrate on their medical duties.

In the following section, we discuss the benefits of workflows for deriving consent policies. The rest of this paper is organized as follows. In Section 3, the case study used through out the paper is introduced. Section 4 provides an overview of our consent-based framework. Section 5 describes the workflow procedures that are used for implementing the case study. In Section 6, the enforcement model of consent policies is outlined. The implementation of the framework, and the specification and enforcement of consent policies is described in Section 7. Related work is discussed in Section 8. We conclude with Section 9 proposing some future research questions.

2 Why Workflow?

One of the significant benefits of using workflow is that it provides better process control. Users must follow predefined procedures, ensuring that the work is performed in the planned way and meets business and regulatory requirements. Workflows can also provide feedback to carers. For instance, if an action defined in a medical procedure is not performed within a certain amount of time then an alarm could be raised. The use of workflows enables the logging of the actions being executed by subjects. These logs can be used for assessing and improving the performance of carers. Moreover, logs should be made available to patients, or they representatives, in case they want to audit the medical procedures to which they were subjected and track responsibilities of subjects in case of negative experiences during their treatment.

Obtaining patient consent can be easily captured as a mandatory step in a medical workflow before any use or disclosure of patient healthcare information. By executing the workflow, the control requirements for patient consent can be enforced.

A workflow also provides a way of limiting access permissions to the context in which an action is being performed enforcing the need-to-know principle. The intuition behind this is that subjects need to access specific parts of the patient's medical record only when they are executing a specific task in a workflow. Therefore by associating permissions with tasks and tracking the execution of tasks, we can ensure that the subjects can access medical records only when they have a need.

Another reason why we consider workflow is that many efforts has been made in developing and experimenting with automated or semi-automated medical workflow systems which support evidence-based medical procedures, therapies and hospital administrations [7, 18, 19, 14]. It is likely that workflow systems will become a core component in future healthcare systems [13].

In our work, we assume that governmental organisations such as the National Health Service in the United Kingdom are responsible for designing and providing to hospitals standard medical procedures. Workflow systems help with this through secure auditing mechanisms that can track and record accesses, and support analysis of anomalies, failure etc.

3 Case Study

In this section, we describe a case study based on the scenario introduced in [9].

Bob Arkwright visits his General Practitioner (GP), Dr Zimmer, because of heart problems. Dr Zimmer performs the basic procedure for heart problems and accesses Bob's record for annotating Bob's symptoms and updating his medical history with his heart conditions. Following, Dr Zimmer believes that Bob heart problems are related to a coronary artery disorder. Dr Zimmer decides that Bob should go to the local hospital to see a specialist.

After Bob is admitted at the hospital, he proceeds to see Dr Hassan, the cardiologist in the local hospital. Dr Hassan decides that Bob needs a radial artery catheterisation (RAC) procedure performed. Bob is taken to the Cardiac Catheterisation Laboratory for the procedure. The results of the test are sent to the ancillary department, where Dr Hassan reviews them electronically together with Bob. Given the location and severity of the artery blockage, Dr Hassan recommends Bob to undergo a Coronary Artery Bypass Surgery to bypass the occluded arteries and create new routes for blood to flow to the heart muscle. Bob agrees to have the procedure performed and is admitted to the inpatient facility.

A surgical team is assigned to Bob. The head of the team, Dr Green, reviews the bypass procedure with Bob along with the anticipated recovery process and time period. Afterwards, Dr Green assigns to Bob a recovery procedure that nurses in the Surgical Intensive Care Unit (SICU) will follow. The recovery procedure prescribes the necessary post-operative medications for Bob.

Bob is aware that Sara, his neighbor, works in the local hospital as surgical nurse and she is part of his surgical team. He decides that she should not be allowed to have access to his medical record. When the team reviews Bob's record, Sara is not allowed to access the record and notifies Dr Green. Because Bob's procedure is very complicated, to safeguard Bob's health Dr Green prefers that all team must be able to access Bob's record in case complications would arise during the surgery. Therefore, he decides to discharge Sara from the team and finds a valid substitute that has access to Bob's record.

During surgery, abnormal liver parameters are found and the surgical team needs to access Bob's record to try to find some useful information about his liver. Bob's medical data related to liver shows that he has a history of alcohol abuse. However, because the complete team has access to his record, they are able to overcome the problems and conclude the procedure successfully.

When Bob fully recovers from the operation, Dr Hassan and Dr Green review Bob's case and medical data and decide that the success of the procedure should be made available to the research community by publishing the results in a major medical journal. However, since the publication of Bob's case (and medical data) is not directly concerned with Bob's health, Dr Hassan and Dr Green should ask an explicit consent to Bob for accessing his record after the procedure is concluded. Bob is notified that Dr Hassan and Dr Green want to access his record for research purposes. He is contacted by them for further explanations and set up a meeting at the hospital. During the meeting, Dr Green and Dr Hassan explain to him that publishing this case could save the life of other people. Moreover, although his case would be made public his identity is not disclosed. However, Dr Green and Dr Hassan ask whether Dr Carter, an Intern Surgeon, could have access to his record as well. In fact, both physicians explain that in the coming months they are going to be very busy and it would be more efficient if Dr Carter could join them in preparing the article.

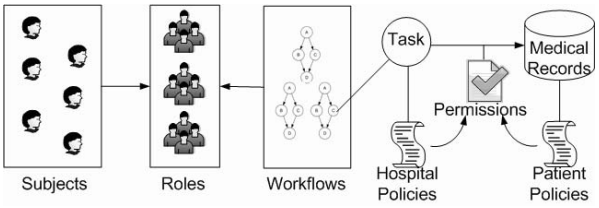

Figure 1. Overview of our Consent-based Framework

Bob agrees on the publication and signs a form that gives his explicit consent to Dr Hassan and Dr Green, although he refuses to provide Dr Carter access to his record. Additionally, Bob specifies the duration of his consent: his record will be accessible to Dr Hassan and Dr Green for a limited period of one month. During the next month, a paper is prepared from his case and it is published in a well-known medical journal.

4 Consent-based Framework

Our workflow based framework consists of the following components: subject, role, workflow, task, policy, permission and medical records (see Figure 1).

Each **workflow** is defined according to a medical protocol or administrative process. It consists of multiple **tasks**. Each task is associated with a set of **policies**. The policies defines the **constraints** for executing the task. Policies can be defined by both the hospital and the patient.

A **subject** is an entity who needs to access a medical record, e.g. a doctor, a nurse. Each subject has a set of attributes. The attributes can be used for authorisation decisions.

A **role** is a named collection of subjects. Unlike in

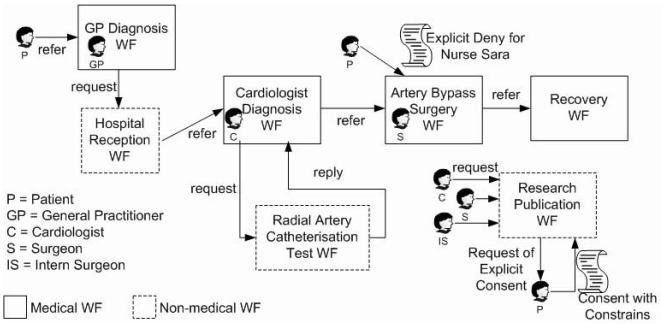

Figure 2. An overview of the workflows involved in the case study.

RBAC, which assign permissions directly to roles, in our framework roles are associated with a set of workflows.

Permissions define the access right the subject has on the medical records when they are executing this task. The permissions define what actions can be operated on which part of a medical record by the subject who is executing the task. Permissions are derived from the enforcement of the policies defined on the task.

A **medical record** is the container of a patient's healthcare information. Most current electronic medical record standards define hierarchical substructures which helps organising the information and make it possible for us to define fine-grained permissions on these substructures.

5 Implementing the Case Study using Workflows

In this section, we discuss the specification of workflows that implement the case study.

The procedures presented in the scenario are represented in our framework as workflows. An overview of the scenario in terms of workflows and subjects is given in Figure 2. In our framework, we distinguish between **medical workflows**, that is workflows implementing medical procedures, from **non-medical workflows**, that are procedures not directly related to the patient's health. In Figure 2, medical workflows, such as the GP and Cardiologist diagnosis, and the surgery, are represented as solid-line blocks. Non-medical workflows, such as the hospital receptionist, the lab test and the preparation of the paper, are represented as dashed-line blocks.

A medical workflow is started by means of a **refer** operation. By using a refer operation, one or more subjects are designated to execute the medical workflow. For instance, in the scenario shown in Figure 2, The patient refers his GP to execute a diagnosis workflow. Non-medical workflows are started as a response to a **request** operation sent by either a subject (i.e. the physicians requiring the starting of the workflow for the publication of the results) or a task in a workflow (such as the task in the cardiologist's workflow that requests the lab tests).

This distinction allows us to associate with each type of workflow a different type of consent policy for accessing the medical record. In particular, the *implicit consent with explicit deny* policy is associated with medical workflows. According to this policy, the patient implicitly consents to a subject executing a medical workflow to access her/his medical record. The subject must provide information on the part of the record that is going to be accessed and the reasons of the access. The patient can however decide to explicitly deny access to any subjects at any time. The rationale behind this policy is that in general a patient accepts to surrender her privacy for the sake of her health. However, in certain cases and to the patient's discretion, the patient can explicitly deny access to one or more subjects. For instance, the patient had a negative experience with a particular physician, or as it is in the case for nurse Sara, the subject is a patient's acquaintance to whom the patient does not wish to provide his personal information.

As for non-medical workflows, they are associated with an *implicit deny with explicit consent* policy. This policy imposes that any subjects executing a non-medical workflow are allowed to access a patient's medical record. The consent should be asked by the subjects as a task in the workflow. Once the consent is given by the patient, then the workflow execution can proceed. Additionally, the patient can specify temporal and validity constraints to restrict the access to the medical record for a fixed amount of time.

In the following, we provide more details on how these consent policies are implemented in our framework. However, before diving into the details of the consent policies, we shall give more details on our model for policy enforcement.

6 Policy Enforcement Model

In our model, a permission to access a patient's medical record is granted by combining and enforcing two separate sets of policies. The first set contains medical policies that are defined and enforced within the medical institutions where the patient is being treated (i.e., hospitals, sheltered houses for elderly people, GP and dentist studios). Policies in this set can be used for defining constraints related to time and location (i.e., a doctor can perform a task only in a specific location during working hours) or to address separation-of-duty (i.e., a doctor prescribing a medication should not be also the pharmacist that sells that medication). The second set of policies contains policies specified by the patients for protecting the privacy of the information stored in their medical records.

When a subject executes a task that requires access to the medical record of a patient, the relevant policies are chosen from each set and enforced accordingly. The overall access control mechanism can be thought of as a two-step process and is represented in Figure 3. In step (1), the Policy Enforcement Point (PEP) enforces authorisation policies defined in the subject's institution (in this case a hospital). If authorisation is granted then the access is evaluated against the set of policies defined by the patient that are enforced by the PEP in step (2). If permission is also granted here, then in step (3) it is possible to get access to the actual data in the medical record.

In this paper we are focusing on the specification and enforcement of patient privacy policies. Therefore in the following discussion, we provide more details on the second step of the access control mechanism.

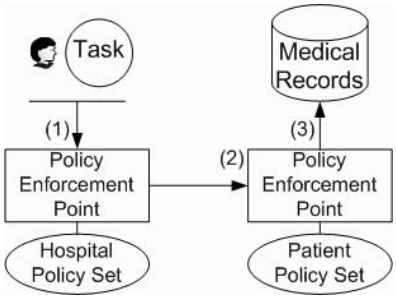

Figure 3. Steps executed for granting a permission.

7 Framework Implementation

The implementation of our framework is realised by integrating two main systems that are described in the following.

7.1 YAWL Workflow System

We use the YAWL system for the specification and enactment of workflows [12]. YAWL provides a very powerful workflow language together with a workflow execution engine, and an editor for creating workflow specifications. YAWL can be customised to export to external components certain events that occur in the life-cycle of workflow instances. On receiving a task-enabled event, a component may decide to 'check-out' the task from the engine. On doing so, the engine marks the task as *executing* and effectively passes operational control for the task to the component. When the component has finished executing the task, it will check it back in to the engine, at which point the engine will mark the task as *completed*, and proceed with the workflow execution.

It should be realised that our framework is independent of the specific workflow language/engine used as long as the workflow system provides means for interacting with our framework.

7.2 Ponder Access Control

The other component in our framework is the policy-based access control module based on the Ponder policy language and interpreter developed at Imperial College London [3]. The language supports the specification of policies for governing the choices in the behaviour of a system [22]. Ponder supports the specification of authorisation policies and event-condition-action (ECA) policies . The policy interpreter organises the entities and resources on which policies operate in hierarchical domains of managed objects. A managed object has a management interface that the object has to implement in order to be managed by the interpreter. Domains allow the classification and grouping of managed objects in a hierarchy. Furthermore, domain paths can be used to address managed objects in policy specifications. Domains can be used to group resources (e.g., data repositories, printers, X-ray machines, etc.), devices (e.g., sensors), and people (i.e., nurses, doctors, GPs, etc).

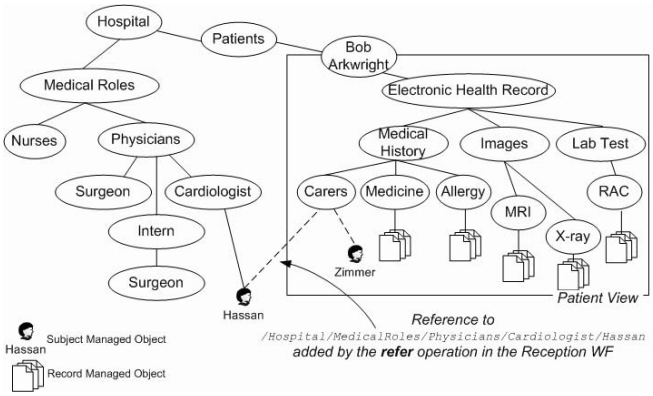

Figure 4. Representation of the case study using Ponder domain structure.

Figure 4 shows how Ponder's domain structure may be used for organising the entities and medical data of the case study. In particular, the domain structure represents an hypothetical hospital where Bob Arkwright is being visited by a cardiologist. Hospital personnel are represented by means of **Subject Managed Object**s (SMO) that could be assigned to the respective domain according to the subject's role. An SMO can be thought of as an electronic credential that identifies a specific individual working in the hospital. The Electronic Health Record (EHR) of a patient, that

is the digital representation of a patient medical record, is also represented as a domain structure. The leaves in this domain structure are **Record Managed Object**s (RMO). RMOs represent specific instances of medical documents, such as the record of allergies, the list of current and past medications, an MRI picture, and so on. It should be noted that, although Figure 4 shows Bob's EHR within the structure of the hospital that Bob is visiting, the actual storage of the EHR itself could be done in a remote location [1]. Domains in Ponder transparently supports external links to point to domains and managed objects contained in domain structures residing in remote location.

The EHR represents the patient's *private view* of her digitalised medical information (shown as a square around Bob's EHR in Figure 4). As such, in our framework patients are enabled to specify access control policies that govern the access to such resources. In Ponder, authorisation policies are used for controlling the rights that entities have on the resources managed in a domain structure. In our case, entities are medical personnel and resources are the patient's medical data, represented as SMOs and RMOs, respectively. Authorisation policies are defined on (subject,target,action)-triples, where the subject is an SMO that executes an action on the target RMO. The language also supports negative authorisation policies, that when applied negate the execution of the defined action. To be able to specify authorisation policies, patients need to refer to the subjects executing the accesses. For this reason, the domain **Carers** in the patient's EHR is used for containing references to SMOs, as shown in Figure 4. This domain contains a collection of references to SMOs of medical personnel known to the patient. For instance, if we consider our case study, when Bob goes to his GP, Dr Zimmer, the Carers domain would contain just the reference to his GP's SMO. When Bob is admitted in the hospital, the receptionist uses a workflow to *refer* Bob to Dr Hassan, resulting in a reference to Dr Hassan's SMO to be added to Bob's Carer domain (shown as a dotted line in Figure 4). The reference is added in terms of the context in which the refer operation is executed. Dr Hassan may have several roles within the hospital and other institutions. This would result in his SMO being contained in different domains. However, the context in which the refer operation is executed defines that a cardiologist within the hospital is required. Therefore, the path that is added to the Carers domain provides a reference to Dr Hassan's SMO as a cardiologist.

Now Bob knows who is his cardiologist and may define authorisation policies specifically for Dr Hassan. Actually, this domain represents an important part of the medical history of the patient because it provides an overview of the medical personnel that have provided care to a patient.

[1]The EHR structure can also be a representation of distributed resources. For instance, RMOs could be stored within the organisations that created them, i.e. the X-ray images in Bob's EHR could be stored in another hospital where Bob attended when he broke is leg.

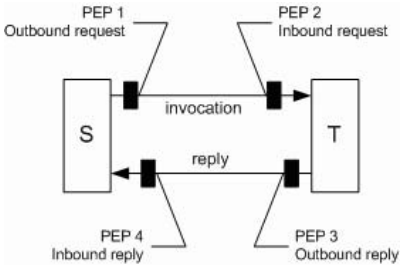

Figure 5. The fine-grained access model supported in Ponder.

An important feature of the authorisation model in Ponder is the fine-grained access control mechanism. Authorisation policies can be independently specified for controlling the subject-side and the target-side of an action. A full description of the model can be found in [20]. Here, for brevity reasons, we just recall the details of the model that are relevant for our discussion.

As shown in Figure 5, the access control mechanism provides 4 different PEPes to enforce policies. Policies can be specific for the subject and the target side of a request. The policies enforced at PEP 1 control the subject when it sends out a request. We name such policies *Subject authorisation (SA)* policies. The PEP 2 is used for enforcing authorisation policies for control on the target side. We name these policies *Target authorisation (TA)* policies. The policies enforced at PEP 4 and PEP 3 are the dual of SA and TA policies. These policies, called respectively *Subject-return authorisation (SRA)* and *Target-return authorisation (TRA)*, are used for controlling the return part of an action. SRA policies can be used for protecting the integrity of a subject (i.e. checking that the reply does not contain malicious data). TRA policies can be used for filtering the data that is returned to a subject.

Using SA and TA policies (enforced at PEP 1 and PEP 2), it becomes possible to employ this mechanism for implementing the model described in section 6. In particular, hospital policies are specified and enforced as SA policies. On the other hand, patients can use TA policies for controlling the access to the private view of their medical data. A patient uses the references contained in the Carers domain for specifying which SMO a TA policy is to be applied.

In case of authorisation conflicts, that could happen when authorisation policies of different signs apply to the same triple, the interpreter is also able to autonomously resolve those conflicts. For more details on the rules that are applied for resolving conflicts, we refer the interested reader to [20]

Ponder ECA policies are used to dynamically adapt the system to changes of either context or behaviour of applications. Events are trigged by such changes and are propagated using an event bus. ECA polices capture events and execute actions for adapting the system. For example, ECA policies can change the domain structure adding/removing domains and managed objects, can invoke action on managed objects, can enable/disable other policies and can trigger other ECA policies by sending more events.

The following sections, we discuss how patient consent policies are enforced in our framework.

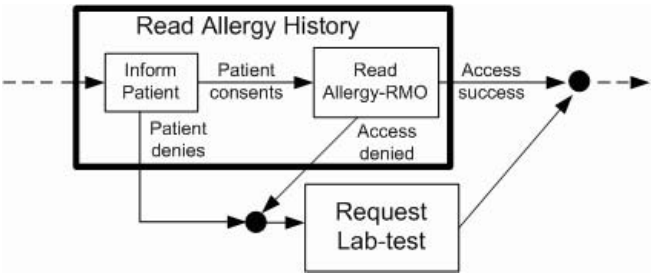

Figure 6. Division in sub-tasks of a task that performs an access to the EHR.

7.3 Consent Meta Policies

In this section, we take a closer look at the enforcement of patient consent policies. In our framework, the contextual information obtained by the workflow execution together with the enforcement of TA and ECA policies are used to derive what we call **consent meta policies**. In the following, we provide more details on two types of consent meta policies that are suitable for medical environments.

7.3.1 Implicit Consent with Explicit Deny Policy (ICED)

The ICED policy is associated with the execution of medical workflows. The main idea behind this policy is that a patient implicitly allows a subject executing a workflow concerning her health to access the EHR. However, the patient can explicitly deny access to one or more subjects involved in the current or any future workflows.

Implicitly allowing subjects executing a medical workflow to access an EHR means that the patient is not required to explicitly specify permissions for the subjects. The system can derive the permissions from the context in which the access is being executed and automatically generate the required permissions. In this way, the **need-to-know** principle is enforced without giving unnecessary burdens to the patients. However, a subject executing a medical workflow should provide information to a patient on each access to the EHR. This means that before the access is performed the subject should explain to the patient which part of the EHR needs to be accessed and the motivations that justify such an access. If the patient thinks that the given motivations are not exhaustive or that more information than necessary is being accessed, the consent can be withdrawn.

In our framework, the enforcement of the ICED policy is achieved using the following two mechanisms.

In order to inform a patient that an access is going to be executed, the task that represents the access is divided into two sub-tasks. Figure 6 shows the splitting of a "Read Allergy History" task in two sub-tasks: "Inform Patient" and "Read Allergy RMO". The first sub-task requires the subject to inform the patient that an access to the allergy record is going to be executed. If the patient consents then the second sub-task is executed, that is the actual access to the record. However, if the patient denies the access, then alternative tasks need to be executed. For instance, in the case of Figure 6 the subject can require a lab test to get the information that is required to carry on the workflow execution. It might be the case that, even if the patient gives the consent, pre-existing negative policies could deny the access. This situation can be captured as an alternative execution of the workflow. In the case shown in Figure 6, if access is denied, then a lab test is requested. If no alternative solutions are available, then the subject executing the workflow can be notified by the workflow engine to manually take some actions. Related research investigating specification and handling of exceptional situations in medical workflows be found in Han *et al.* [16].

The automatic provision of access rights is performed by means of Ponder ECA policies. Figure 7 provides an overview of the generation of such permissions. The example shown is that of Dr Hassan that wants to read the allergy history of Bob. After Dr Hassan obtains Bob's consent to proceed, the "Read Allergy RMO" sub-task is declared *executing* by the YAWL workflow engine and an event with this information is sent. The event also carries context-dependent data, such as: the specific instance of the task that is activated, that the task is part of a medical workflow, the subject that is executing it, the target and action that is performed, the location and time. This event triggers an ECA policy, that using the data carried by the event, generates and activates a positive TA policy. This policy, called +TA1, is shown in Figure 7 as an arrow connecting Dr Hassan's SMO to the AllergyRMO. When Dr Hassan, via his SMO, performs the action of reading the AllergyRMO, the access control mechanism captures such an action and enforces authorisation policies at PEP1 and PEP2. The SA policies enforced at PEP1 are specified by the hospital. We assume that Dr Hassan complies with the hospital policies and that from the point of view of the hospital the action is

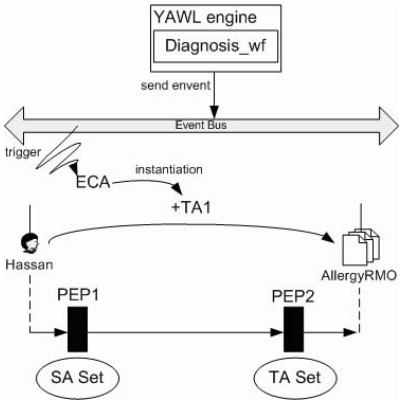

Figure 7. Representation of the TA policy generated by an ECA policy.

allowed. Another enforcement of authorisation policies is done at PEP2. Here, the mechanism searched for TA policies. Because policy +TA1 is already in place the action can be authorised. (Although, it could be the case that there are conflicts with other policies previously specified by Bob; we discuss this case later in this section). Another important step that is not shown in Figure 7 is that of automatically disabling the authorisation policy. After this sub-task is concluded, the YAWL engine sends an event that triggers another ECA policy. This ECA policy will take care of disabling policy +TA1. This use of ECA policies guarantees that the permission for accessing the EHR is available only for the execution of the specific task.

Our framework enables a patient to define negative authorisation policies that deny subjects access to the EHR. Once the SMO reference becomes available into the Carer domain in the EHR, the patient can define a negative TA policy for that specific SMO. When the system generates the positive TA using the mechanism describe above, this positive TA will be in conflict with the negative TA created by the patient. In this case, the conflict resolution mechanism will give priority to the patient's negative TA policy. This means that the access is denied and an alternative action must be taken. For instance, in our case study, the surgeon is notifies that a nurse (Sara) is not allowed to access Bob's EHR when the surgery team is executing the debriefing task in the Artery Bypass Surgery workflow. Therefore, before executing the actual surgery, the surgeon decides to remove the nurse from the team.

7.3.2 Implicit Deny with Explicit Consent Policy (IDEC)

The IDEC policy is used for non-medical workflows. This type of policy implies that no access to the patient EHR is given to subjects executing non-medical workflows. If a SMO requires access to the EHR during a non-medical workflow the consent must be explicitly asked to the patient. It is up to the patient to decide whether or not to give the required consent.

The patient can decide to give the consent by specifying a positive TA policy. The policy must be as restrictive and specific as possible. The subject should be the specific SMO of the subject and the target the necessary record. Moreover, the action can only be a "read" because a subject executing a non-medical workflow must not be allowed to change any part of an EHR. The policy should be associated with an expiring time (and/or a number of access for which is valid) and could restrict the time and location where the consent is granted.

8 Related Work

The BMA policy model was developed in late 1990s in response to the National Health Service (NHS) project of building a nationwide medical database. The access privileges for each medical record are defined in the form of access control lists (ACLs) and managed by a responsible clinician who is the only one can change the ACLs. The main goal of the BMA model is to enforce the principle of patient consent, and to prevent too many people getting access to too large databases of identifiable records. A prototype implementation has been built in a General Practice environment [17]. Apart from the weakness we discussed in the introduction, other criticisms include the ACLs are not flexible and expressive enough, and the model is more clinician-centered rather than patient-centered.

Cassandra is a role-based trust management language and system for expressing authorisation policy. Cassandra supports credential-based authorisation between administrative domains, and rules can refer to remote policies for credential retrieval and trust negotiation. The main focus of Cassandra are healthcare systems. Cassandra has been used to develop a policy for the UK national electronic health record (EHR) system, based on the requirements of the NHS' National Programme for Information Technology (NPfIT). The notion of consent is captures in Cassandra as a special role. The consent is given to a subject, that is assigned to a special role, only if the subject requests the consent from the patient. This requirement adds extra workload for the healthcare professionals. As we shown in this paper, in most situations the consent can be implicitly derived from the context.

Also relevant to our discussion is the RBAC model proposed in [15, 21]. RBAC is motivated by the observation that in the real-world most access control decisions are based on the subject's job functions in an organisation. This observation is valid for organisations that own the data that

is being accessed. For healthcare organizations however, the medical data that is being accessed is "owned" by patients. Ideally, medical data should only be disclosed when consent is obtained from patients. But the notion of consent is not captured by the original RBAC model. Thereby, extensions to this model are required for its applicability in healthcare systems (such as Cassandra).

The idea of using the workflow concept for deriving access rights was first conceived by Atluri and Huang in [8], where they introduce the Workflow Authorisation Model (WAM). In WAM, authorisation constraints for data and resources are synchronised with the execution of workflows. Our framework described here is a refinement of the WAM model where the patient consent plays a crucial role in the access control decisions.

9 Conclusion and Future Work

In this paper, we presented a framework for enforcing consent policies for healthcare systems based on workflows. In our framework, patients' consent has a central role for assigning permissions to subjects that access patients' medical data. The context that is derived from the workflow execution is used for enforcing consent policies. Patients can fine tune those policies and effectively control the subjects to which consent is given or withdrawn. Additionally, the use of workflows allows us to enforce the need-to-know principle whereby a subject can access the patient's medical data only if there is a specific need. In our framework, this need is associated with the execution of specific workflows.

As part of our future work, we intend to carry out performance measurements of our prototype. Another area that we want to explore is that of using our framework for expressing meta policies to capture *conflict-of-interest* and *separation-of-duties* policies and to apply the framework to in financial applications.

Acknowledgments

This research was supported by the UK's EPSRC research grant EP/C537181/1 and forms part of the CareGrid, a collaborative project with the University of Cambridge. The authors would like to thank the members of the Policy Research Group at Imperial College for their support.

References

[1] European standards on confidentiality and privacy in healthcare. www.eurosocap.org.

[2] HHS Standards for Privacy of Individually Identifiable Health Information, US Code of Federal Regulations: 45 CFR § 160.103.

[3] The ponder2 project. www.ponder2.net.

[4] Protecting privacy in computerized medical information. Office of Technology Assessment, US government printing office, 1993.

[5] R. J. Anderson. A security policy model for clinical information systems. In *SP '96: Proceedings of the 1996 IEEE Symposium on Security and Privacy*, page 30, Washington, DC, USA, 1996. IEEE Computer Society.

[6] R. J. Anderson. An update on the bma security policy, 1996.

[7] L. Ardissono, A. D. Leva, G. Petrone, M. Segnan, and M. Sonnessa. Adaptive medical workflow management for a context-dependent home healthcare assistance service. *Electr. Notes Theor. Comput. Sci.*, 146(1):59–68, 2006.

[8] V. Atluri and W. kuang Huang. An authorization model for workflows. In *ESORICS*, pages 44–64, 1996.

[9] M. Y. Becker. *Cassandra: flexible trust management and its application to electronic health records*. PhD thesis, University of Cambridge, October 2005.

[10] M. Y. Becker and P. Sewell. Cassandra: Distributed access control policies with tunable expressiveness. In *POLICY*, pages 159–168. IEEE Computer Society, 2004.

[11] M. Y. Becker and P. Sewell. Cassandra: Flexible trust management, applied to electronic health records. In *CSFW*, pages 139–154. IEEE Computer Society, 2004.

[12] W. V. der Aalst, L. Aldred, M. Dumas, and A. ter. Design and implementation of the yawl system, 2004.

[13] A. Dwivedi, R. Bali, A. James, and R. Naguib. Workflow management systems: the healthcare technology of the future? In *the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 4, pages 3887–3890, 2001.

[14] S. W. T. et al. Modeling guidelines for integration into clinical workflow. In *Medinfo*, pages 174–178, 2004.

[15] D. Ferraiolo and R. Kuhn. Role-based access controls. In *15th NIST-NCSC National Computer Security Conference*, pages 554–563, 1992.

[16] M. Han, T. Thiery, and X. Song. Managing exceptions in the medical workflow systems. In *ICSE '06: Proceeding of the 28th international conference on Software engineering*, pages 741–750, New York, NY, USA, 2006. ACM.

[17] A. Hassey and M. Wells. Clinical systems security – implementing the bma policy and guidelines. In R. Anderson, editor, *Personal Medical Information – Security, Engineering and Ethics*, pages 79–94. Springer-Verlag.

[18] M. Poulymenopoulou and G. Vassilacopoulos. A web-based workflow system for emergency healthcare. In *Medical Informatics Europe*, 2002.

[19] S. Quaglini, M. Stefanelli, G. Lanzola, V. Caporusso, and S. Panzarasa. Flexible guideline-based patient careflow systems. *Artificial Intelligence in Medicine*, 22(1):65–80, 2001.

[20] G. Russello, C. Dong, and N. Dulay. Authorisation and conflict resolution for hierarchical domains. In *POLICY '07: Proceedings of the Eighth IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 201–210, Washington, DC, USA, 2007. IEEE Computer Society.

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

[22] M. Sloman and E. Lupu. Security and management policy specification. *IEEE Network*, 16(2):10–19, March 2002.