Privacy-Aware Workflow Management

Bandar Alhaqbani[1], Michael Adams[2], Colin Fidge[2], and
Arthur H.M. ter Hofstede[2]

Faculty of Science and Technology, Queensland University of Technology,
GPO Box 2434, Brisbane QLD 4001, Australia
[1]b.alhaqbani@isi.qut.edu.au
[2]{mj.adams,c.fidge,a.terhofstede}@qut.edu.au

Abstract. Information security policies play an important role in achieving information security. Confidentiality, Integrity, and Availability are classic information security goals attained by enforcing appropriate security policies. Workflow Management Systems (WfMSs) also benefit from inclusion of these policies to maintain the security of business-critical data. However, in typical WfMSs these policies are designed to enforce the organisation's security requirements but do not consider other external security requirements. Privacy is an important security requirement that concerns the subject of data held by an organisation. WfMSs often process sensitive data related to subjects who demand that their data is properly protected, but WfMSs fail to enforce the subjects' privacy desire due to their inability to capture and enforce privacy policies. In this paper, we illustrate existing WfMS privacy weaknesses and introduce the WfMS extensions required to enforce data privacy. We implemented the extensions in the YAWL WfMS environment and present a case study to demonstrate how our extended WfMS enforces a subject's privacy policy.

1 Introduction

Information security rests on confidentiality, integrity, and availability. These 'hard' security requirements are considered carefully when developing and using information systems. However, 'soft' security requirements, e.g. privacy and trust, are neglected. In most cases, information security requirements are set to satisfy an organisation's security policy, but not those of other stakeholders, most notably the individuals and institutions who are the subjects of the data held by an organisation.

Privacy is a crucial security requirement that concerns users participating in e-business processes. In response to this concern, governments have set privacy laws, e.g. Australia's Privacy Act and the U.S. Health Insurance Portability and Accountability Act (HIPAA). According to Alan Westin [27], "Privacy is the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others". However, there is a popular misconception that data confidentiality processes adequately cover data privacy requirements. In fact, traditional data confidentiality mechanisms aim to give the *owner* of data control over its accessibility, whereas privacy means giving the *subject* of data control over who accesses it.

In the information system arena, Workflow Management Systems (WfMSs) are used to run day-to-day applications in numerous domains. A workflow separates the various activities of a given organisational process into a set of well-defined tasks. The tasks are executed according to the *organisation*'s policies to achieve certain objectives. Among these policies, security policies are crucial for ensuring that the organisation is adhering to its hard security objectives. However, many workflows deal with different types of data that originate from various sources. Once the data is retrieved for a particular workflow case, the organisation, through its WfMS, is responsible for maintaining data confidentiality as per the organisation's confidentiality policy. Well-crafted workflow access control mechanisms help the organisation to attain its security objectives by assigning tasks' execution to authorised (human) resources only.

However, the workflow system might also hold descriptive data about a user which, in the user's opinion, would cause a privacy violation if it was accessed by certain workflow authorised resources. In order to execute the workflow case securely, and satisfy the user's privacy wishes, we need to consider the user's privacy policy, that states the user's access authorisation, in the workflow access control mechanism. Current WfMS structures do not provide a way to capture the user's privacy wishes because they fail to identify and respect the wishes of the workflow's *subject*. Currently proposed workflow access control models are built with only the organisation's policy in mind and fail to introduce and discuss the workflow *subject*'s privacy requirements.

The subject's privacy policy impacts workflow execution in two ways. On the one hand, it affects the resource allocation process. Usually, this process is implemented according to the organisation's policy. However, the subject's privacy policy acts as a filter that excludes workers not authorised by the subject from the workflow's allocateable resources. On the second hand, it affects data presentation when rendering data forms so that sensitive data is not revealed to unauthorised users. It thus functions to maintain privacy of the subject's data by concealing any sensitive content.

In this paper, we introduce the *subject* notion and its implications into the workflow system's security state especially the *privacy* filtering aspect. This is demonstrated with three examples. In addition, we present a conceptual model which introduces the *subject* notion into the workflow authorisation model. In order to validate our proposed model, we have implemented our conceptual model in the YAWL environment [14], producing a novel secure work-resource allocation strategy with auxiliary data properties which are utilised to control access to private data. Finally, we use a healthcare case study to prove the effectiveness of our implemented approach.

2 Motivating Examples

Workflow subjects and their privacy requirements have not gained sufficient attention when designing and executing workflow models. In this section, we present three examples which highlight the privacy and conflict of interest im-

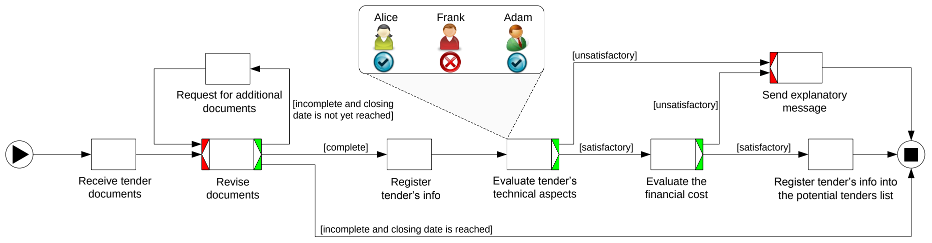

Alice
Frank
Adam
Request for additional documents
[incomplete and closing date is not yet reached]
[unsatisfactory]
[unsatisfactory]
Send explanatory message
[complete]
[satisfactory]
[satisfactory]
Receive tender documents
Revise documents
Register tender's info
Evaluate tender's technical aspects
Evaluate the financial cost
Register tender's info into the potential tenders list
[incomplete and closing date is reached]

Fig. 1. Tender evaluation workflow model

plications that result from neglecting the subject of the workflow in a workflow's design and execution processes. We use the YAWL notation to model and illustrate these examples. Through these examples, we determine the workflow extension requirements that are described in more detail in Section 3.

2.1 Conflict of Interest: Contract Tender Evaluations

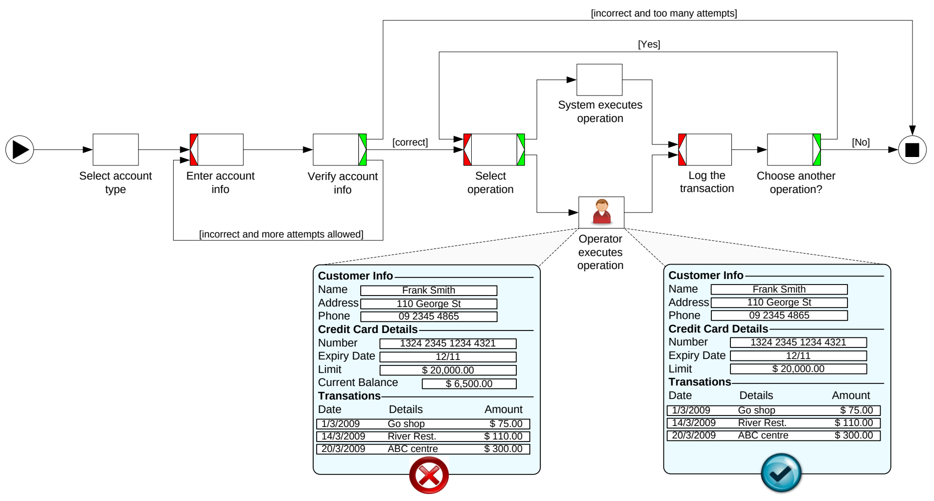
In business, contract tenders are evaluated in several steps as illustrated in Figure 1. The process starts by receiving the tenderer's documents and putting them through technical and financial evaluations. These tasks are allocated to available resources to carry out in a way consistent with the organisation's security authorisation policy.

However, we can identify a security threat in this example that results from not considering the subject of the workflow in the authorisation process. Let's assume that the ACME company has submitted a tender document to a government agency. As per the agency's authorisation policy, either of *Alice*, *Frank*, or *Adam* can perform the technical evaluation for any submitted tender. Let's further assume that *Frank* is a shareholder in ACME and is allocated the technical evaluation task of ACME's tender. As a result, this context creates a conflict of interest which might compromise *Frank*'s actions. This allocation creates a *conflict of interest* with might compromise *Frank*'s actions in a way that does serve the organisation's best interests.

This problem occurs because the organisation can not express an authorisation constraint that excludes those human resources that are in a conflict of interest with the company submitting a tender. Instead, the company's identity should be used as the workflow system's 'subject' property so that the organisation can easily create an authorisation constraint to protect against any conflicts of interest, e.g. due to the evaluator of a tender also being a shareholder in the tendering company.

2.2 Hiding Personal Data: Phone Banking

In the banking sector, phone banking is a useful service that provides substantial benefits. Figure 2 illustrates a phone banking workflow model that receives and


[incorrect and too many attempts]
[Yes]
System executes operation
[correct]
[No]
Select account type
Enter account info
Verify account info
Select operation
Log the transaction
Choose another operation?
[incorrect and more attempts allowed]
Operator executes operation

Customer Info

Name	Frank Smith
Address	110 George St
Phone	09 2345 4865

Credit Card Details

Number	1324 2345 1234 4321
Expiry Date	12/11
Limit	$ 20,000.00
Current Balance	$ 6,500.00

Transations

Date	Details	Amount
1/3/2009	Go shop	$ 75.00
14/3/2009	River Rest.	$ 110.00
20/3/2009	ABC centre	$ 300.00

Customer Info

Name	Frank Smith
Address	110 George St
Phone	09 2345 4865

Credit Card Details

Number	1324 2345 1234 4321
Expiry Date	12/11
Limit	$ 20,000.00

Transations

Date	Details	Amount
1/3/2009	Go shop	$ 75.00
14/3/2009	River Rest.	$ 110.00
20/3/2009	ABC centre	$ 300.00

Fig. 2. Phone banking workflow model

processes customer requests. Requests are processed automatically by the system or manually by an operator. In this particular case, the customer has no control over what can be seen of his bank account's data by the workflow-authorised operator. This is due to data access control being managed by the bank's security policy without consideration of the customer's privacy wishes.

For instance, a customer may wish to *hide* data, such as his credit card balance, from the bank operator when he is enquiring about a suspicious transaction that occurred in his credit card account. Currently, this precise privacy control cannot be implemented in a WfMS. However, extending the workflow system to capture the subject of the workflow would allow it to retrieve the subject's privacy policy. This can then be employed by the workflow engine to allocate the task to an appropriate authorised resource from both the workflow's and the subject's perspective. In addition, the workflow engine could conceal the subject's private data from generated forms visible to unauthorised resources. In Figure 2, for instance, the displayed form should not include the customer's credit card balance.

2.3 Generalising Data: Social Networking

Social networks hold collections of data with different privacy levels. Although they provide various services, vigilance is required in maintaining the privacy of their users [7]. The workflow model example in Figure 3 aims to produce a 'friends' album from the Facebook network. In this process, the user selects friends that he wants to include in his album and then selects the information that he wants to retrieve about them. Before producing the album, the user examines the retrieved information and selects the information that he wants to save.

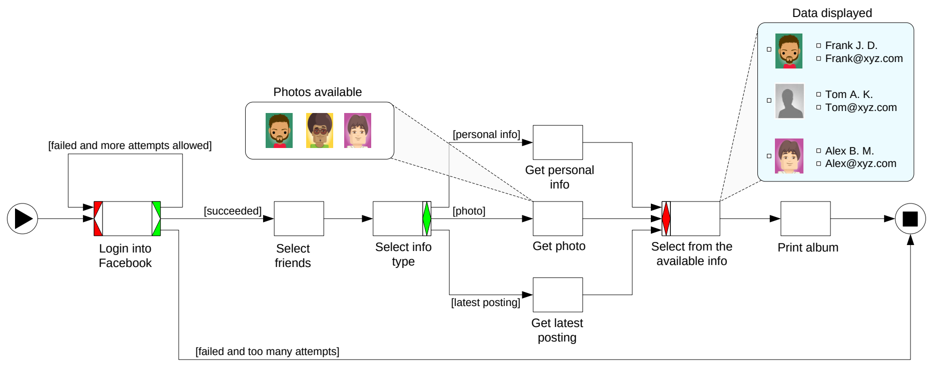


Fig. 3. Create friends album workflow model

In a current WfMS, this process would be executed by retrieving the user's friends', i.e. the workflow's subjects, information by presenting them to the user without considering any filters that the subjects may wish to set. For example *John* has friends in his Facebook network and *Tom* is one of them. However, *Tom* does not want to have his photo in any album that *John* would make. Let's assume that *John* starts the 'create friends album' workflow process and selects his friends, including *Tom*, and requests to have his friends' personal information and photo. The workflow will get this information but, should enforce *Tom*'s privacy wish to withdraw his photo from *John*'s album. To satisfy *Tom*'s privacy desire, the workflow engine should know about the workflow's subjects including *Frank*, *Tom*, and *Alex*, and enforce their privacy policies while executing the workflow case. In this particular situation, the workflow engine could enforce *Tom*'s privacy requirement by *generalising the data* made available to John, by substituting his photo with a generic image as shown in Figure 3.

3 Workflow Implications

The examples in Section 2 illustrated the inability of current WfMS authorisation policies to preserve a workflow subject's privacy and avoid conflicts of interest. In this section, we introduce four technical requirements needed to overcome these security problems and to enhance the workflow authorisation constraints. Also, we present a practical approach for adding these technical requirements to a workflow management system.

3.1 Adding the Subject to Workflow Designs

The workflow design phase defines the workflow specifications that are required to achieve a certain objective (e.g. processing an insurance claim). The workflow specification must include tasks, resources, control flow, and work allocation strategies. The workflow authorisation policy consists of authorisation constraints that are defined by the workflow administrator. For example, the workflow administrator can set a 'separation of duty' security constraint in the execution of tasks T_1 and T_2 so that whoever executes task T_1 should not execute T_2. During the execution of the workflow case, the workflow engine is aware of this constraint and can use the workflow case log file to know who was the executor of task T_1 and avoid assigning task T_2 to that resource.

However, such a workflow authorisation policy cannot capture security constraints related to how a user's or an organisation's data is employed by the workflow case, e.g. the bank customer in the phone banking workflow model (Section 2). In order to strengthen the workflow authorisation policy to protect against threats to privacy, we must utilise the workflow subject's relevant information and privacy requirements in the workflow authorisation policy. Therefore, we must introduce the concept of *subject* to the workflow specification during the workflow design phase.

We define a workflow's subject as an entity that owns some of the workflow's data or is described and identified by this data. The subject is a uniquely identifiable human, e.g. a bank customer in the phone banking workflow (Section 2.2), or an institution, e.g. the tendering company in the tender evaluation workflow (Section 2.1). Also, it can be a single entity, e.g. a bank customer, or several, e.g. all the of user's friends in the 'create friends album' workflow model (Section 2.3). In the workflow design phase, we can then create the subject's related authorisation constraints by using the workflow's subject reference which tells us who a particular data item is about. This reference can then be used by the workflow engine while executing a workflow case to retrieve the subject's relevant information, e.g. the subject's privacy policy, and employ this in its access authorisation mechanism.

3.2 Auxiliary Data Properties for Privacy Requirements

In WfMSs, the workflow data perspective is developed during the design phase where it describes the data that will be manipulated by the workflow case. In order to include privacy properties, we need to capture their definition in the same data perspective. That is, we need a way to introduce data properties in an ad-hoc fashion without editing the primary workflow data.

Here, we introduce auxiliary data properties that are not part of the workflow data structure perspective definition. Auxiliary data properties are metadata descriptors (attribute-value pairs) associated with workflow data elements. Each workflow data element may have a number of auxiliary properties that at runtime may influence certain actions, or change the presentation of data when it is rendered. We can use this data at any stage of the workflow design phase and link it to the workflow data it is associated with. This predefined data is utilised to serve various functions while executing a workflow case. For example, it can be used to specify a more precise data validation error message when the workflow data fails a validation test and the default message is ambiguous. Alternately, during the data form rendering process for a task, the workflow engine may examine its auxiliary data properties for font or background colour definitions, or to render a particular type of component to display the task data.

For privacy purposes, we can utilise such auxiliary data properties in our access control mechanism so that, based on their values, they may influence the workflow engine to protect private data. This can be accomplished by considering the privacy rules in the subject's privacy policy and the resource(s) that will execute a task. Let's use the phone banking example (Section 2.2) to show how the auxiliary data properties are set. *John*, who is a bank customer, has a privacy policy which says that he does not want *Tom*, who is a bank employee, to read his credit card balance when *Tom* handles *John*'s call to the phone banking system. The credit card balance data is linked to an auxiliary data property, let's name it `hide`, that functions to maintain the privacy state of the data. Now, let's assume that *John* calls and *Tom* is the available employee to handle *John*'s task. In this case, the `hide` property will tell the form rendering engine to hide the credit card balance field in order to preserve *John*'s privacy. However, if another employee handles *John*'s task, the credit card balance's `hide` property would not be set and the data will be displayed in the form.

In general, there are two ways to suppress data [22]. To preserve a subject's privacy we therefore need two auxiliary data properties:

1. **The hide property** is an auxiliary data property that is assigned for each data element. It serves to direct the form rendering engine to hide the existence of a private field and is set dynamically according to the subject's privacy policy. This is accomplished by not listing the private data when rendering the form that results from executing a workflow task. Hiding the credit card balance in Section 2.2 is an example of this.
2. **The generalise property** functions similarly except that it instructs the workflow form engine to generalise the data in a way that the resource's knowledge of the private data is minimised. The generalise property maintains the semantics of the generated form whereas the hide property does not. Substituting a generic image in Section 2.3 is an example.

3.3 Privacy-Preserving Work Allocation

Several work allocation patterns have been introduced to accommodate work-resource allocation requirements in workflows [21]. However, these work allocation strategies did not consider the subject of the workflow and thus did not take into account the privacy requirements and the potential conflicts of interest with the subject. As illustrated earlier, the subject's privacy requirements influence the work-resource allocation process to allocate the work item to non-restricted resources from the subject's perspective. Let's recall the phone banking example and assume that *John* has called the system and there are two employees who can take *John*'s request, *Tom* and *Matt*. *Tom* is restricted by *John* from accessing *John*'s credit card balance whereas *Matt* is not. In this case, the WfMS should consider *John*'s privacy policy and preferentially offer the task to *Matt* who is not restricted by *John*.

However, in some cases assigning the task to a non-restricted resource cannot be achieved. For example, let's assume that *John* restricts *Matt* from accessing his credit card transaction details which means that *Matt* should not access any of the three possible transaction events. As a result, *Tom* and *Matt* are both restricted by *John*. To solve the work-resource allocation problem, the workflow management should allocate the task to the resource that has the lowest restriction level. To choose the resource, the workflow management system needs to calculate a restriction weight for each potential resource and then select the resource who has the lowest weight.

3.4 Data Patterns for Private Information

Several workflow data patterns are introduced by Russel et al. [20]. Among these patterns, both workflow data pull and push patterns are required to enhance workflow privacy awareness. Usually the subject's data, which includes his privacy policy, would be stored in an external database that is partially maintained by the subject. In order for the workflow to capture the subject's privacy policy, it needs to retrieve this data from the external database. The workflow data pull pattern is defined as the ability of the workflow to request data elements from resources (e.g. external databases) or services in the operational environment. This pattern bridges the connection between the WfMS and the subject's privacy rules and thus enhances the WfMS's privacy awareness. In addition, the workflow data push pattern allows the workflow to initiate the passing of data elements to a resource (e.g. an external database) or service in the operational environment. This pattern allows the workflow to update the subject's external database and to include any new information, for instance, to alert subjects to attempts to access their private data.

Current WfMSs do not fully support these two data patterns. Their inability to support these patterns implies that contemporary WfMSs cannot capture a subject's privacy policy and enforce it accordingly while executing a workflow case.

4 Conceptualisation

To capture these requirements, we developed a conceptual Object Role (OR) model [13] that addresses the meta data requirements for subjects and their privacy policies for use by WfMSs. For clarity, we partitioned the conceptual model into five parts, where each part concerns a specific concept.

Figure 4 depicts the OR model for our resource concept in a WfMS. Resources come in different types, e.g. `subject` and `employee`. A user can be either a subject, e.g. a bank customer, an employee, e.g. a bank teller, or both. In the organisation structure, the employee occupies one or more job positions that is uniquely identified by an ID (*jobPosition_ID*), e.g. `Tom` holds *jobPosition_ID* `010`. For administrative supervision purposes, the holder of a job position reports to a higher administrative job position, e.g. the holder of job position `010` reports

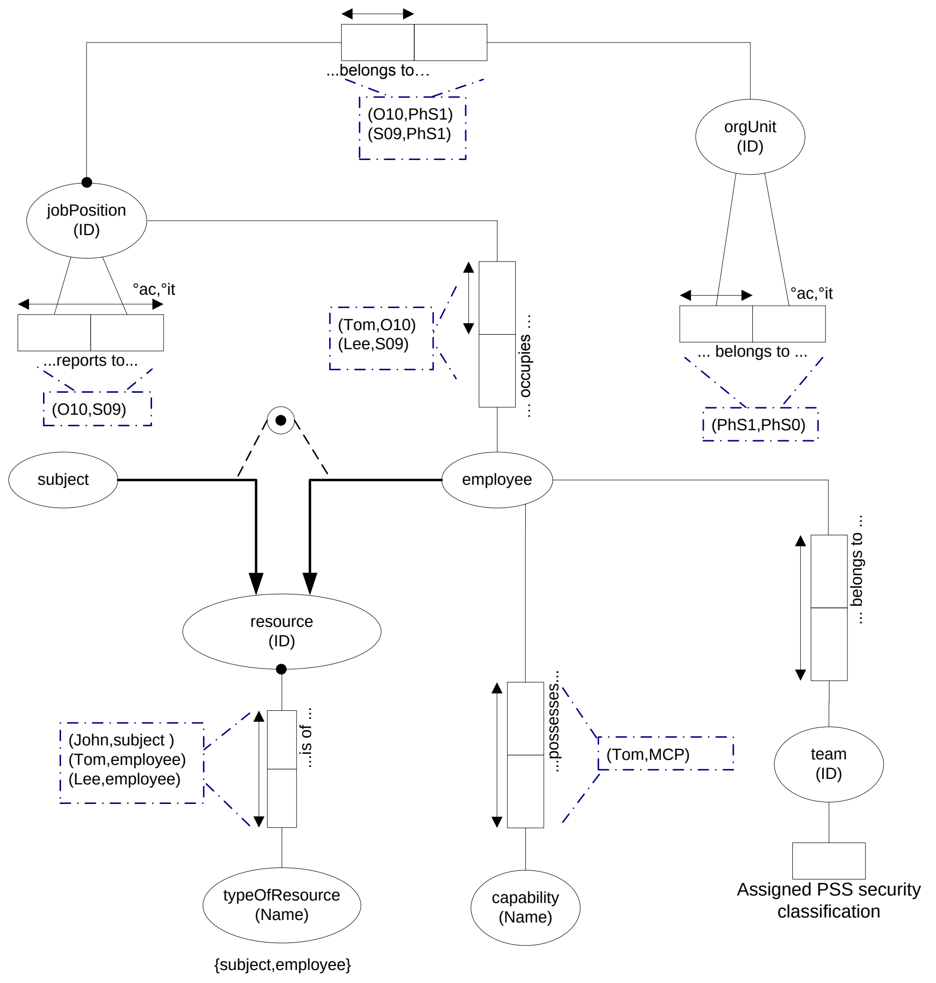


Fig. 4. Conceptual model - resources

to `S09` and to those job positions that supervise `S09` as well. Each job position belongs to a unique organisational unit that is identified by an ID (*orgUnit_ID*), e.g. job position `O10` belongs to organisational unit `PhS1`. Similar to job supervision, each organisational unit administratively belongs to another organisational unit, e.g. organisational unit `PhS1` belongs to `PhS0`. An employee may possess some capabilities that can be used by the WfMS to determine suitable resources to execute a task, e.g. Tom is a Microsoft Certified Professional (`MCP`). In some business cases, there is a requirement to build a team that is responsible to handle a specific task, e.g. a software system's GUI upgrade project team. The team members may have additional privileges to help them proceed in their mission,

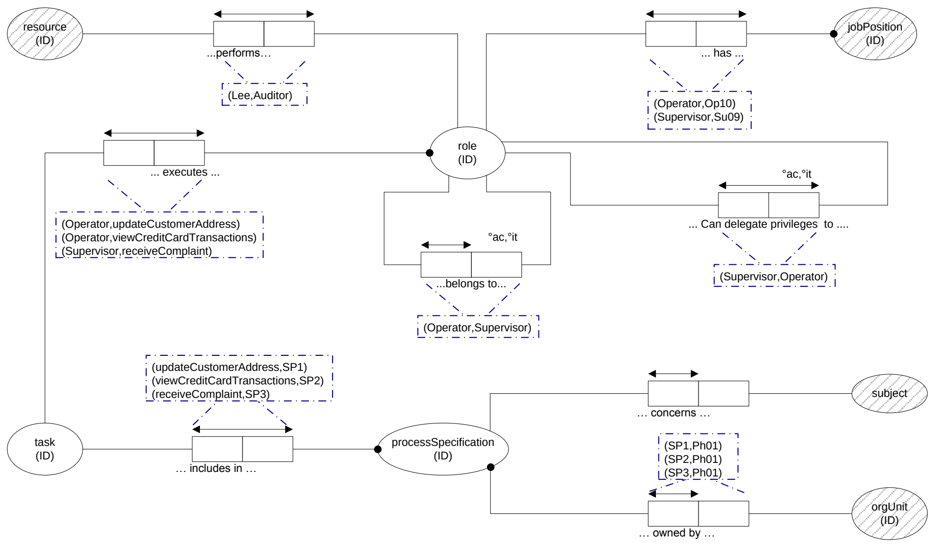


Fig. 5. Conceptual model - roles and tasks

e.g. a higher security classification level to allow them to access the computer room.

The role-task concept in our model, shown in Figure 5, uses some of the entities from the resource model. These entities are represented by a slash fill notation to indicate that these entities are external and have been retrieved from other models. Figure 5 shows the relation between task and role, and role assignment. In WfMS authorisation, Task-Role-Based Access Control (T-RBAC) [19] is widely adopted. T-RBAC is an extension to the well-known Role Based Access Control (RBAC). In T-RBAC, the task notion is inserted between the role and data objects that are introduced in RBAC. In our model, one or more roles can be assigned to a job position, and also a job position can be assigned to many roles. In addition, a resource can be assigned directly to an additional role that is not part of the resource's job position's roles, e.g. `Lee` is assigned an `auditor` role. In order to comply with the role inheritance feature that is introduced in RBAC, in our model a role belongs to one or more parent roles, e.g. the `operator` role belongs to the `supervisor` role. Delegation of authority is a useful feature that allows a user to temporarily transfer his privileges to another user to carry out a specific task on his behalf. In our model, we allow for this feature by specifying which role can be delegated and to which roles this delegation should be given, e.g. the `supervisor` role can be delegated to the `operator` role. Each role can be used in more than one task and a task can be executed by more than one role, e.g. the `updateCustomerTask` can be executed by a resource in the `operator` role. A process specification is a container of tasks, and is defined by a unique identifier, e.g. the `updateCustomerAddress` belongs to process specification `SP1`. Each process specification is owned by an organisational unit, e.g. organisational unit

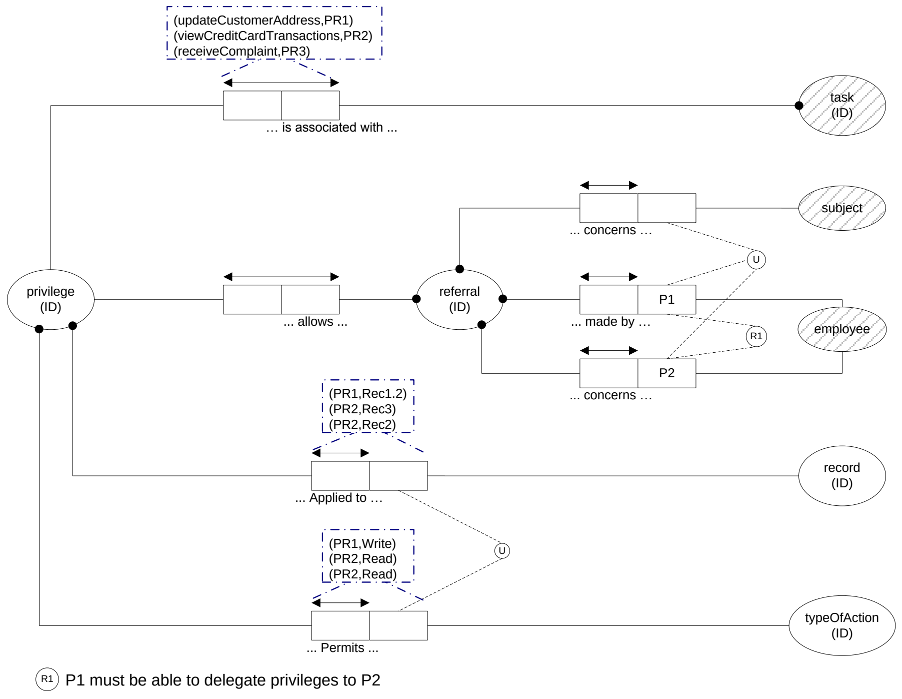


Fig. 6. Conceptual model - privileges

`Ph01` owns process specification `SP1`. The process specification might be about a certain subject, e.g. the `checkCustomerCreditCardDetails` process concerns a specific customer.

In Figure 6 we show the OR model for the privilege concept. A privilege is identified by a unique privilege identifier and has a unique combination of an action and a data record which implies the permitted action is allowed on the data record, e.g. privilege `PR1` permits a `write` action on data record `Rec1.2`. The privilege is assigned to a resource by either linking it to a task, e.g. task `updateCustomerAddress` is associated with privilege `PR1`, or through a referral process. In the referral process, an employee (delegator) refers a subject's case to another employee (delegatee) but with certain privileges. In order to complete this referral successfully, the delegator's role must be allowed to delegate to the delegatee's role which can be checked by looking into the role delegation relation in Figure 5.

The data concept of our model is captured by the OR model in Figure 7. We use a credit card form example in Figure 4 to illustrate how the OR model in Figure 7 is capable of capturing a form's data and its structure. Each record in the OR model is identified by a unique ID. Each record is either a parent of other records or a child. If it is a parent, we capture the record ID and its children's ID and name. For example, record `PersonalInfo` is identified by

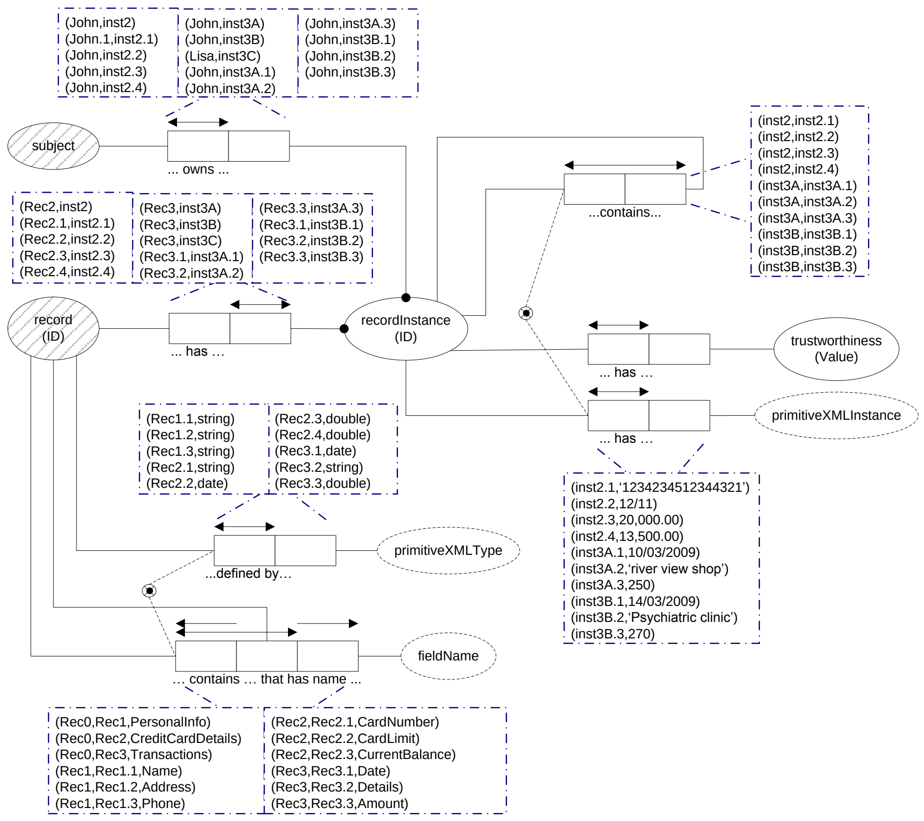
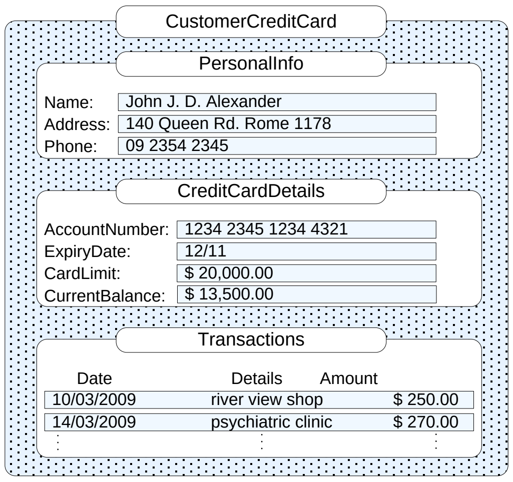


Fig. 7. Conceptual model - data structures

record ID `Rec1` and has three child records. Each child is a record that has a name and unique ID. For example, record ID `Rec1.2` is a child of `Rec1` and has the name `Address`. If a record does not have a child, we capture the record's primitive XML type. For example, `CardNumber` has record ID `Re2.1` and has no child record so we assign the appropriate XML type to it, which is in this case character string. By applying these two relations on the credit card form, we capture the data structure in terms of data relations and XML types as shown in Figure 7. With regard to the data value part, we use a record instance to capture the data value characteristics. Each record instance has a unique ID and must relate to a record and be owned by a subject. For example, record instance `inst2.1` is related to record `Rec1.2` and owned by the subject `John`. The record instance is either a parent of other record instance(s) or a leaf (i.e. childless). For example, in Figure 7 record instance `inst3A` is a parent of record instances `inst3A.1`, `inst3A.2`, and `inst3A.3`. The data value is captured by leaf record instance that corresponds to the a leaf record. For example, record


CustomerCreditCard

PersonalInfo

Name:	John J. D. Alexander
Address:	140 Queen Rd. Rome 1178
Phone:	09 2354 2345

CreditCardDetails

AccountNumber:	1234 2345 1234 4321
ExpiryDate:	12/11
CardLimit:	$ 20,000.00
CurrentBalance:	$ 13,500.00

Transactions

Date	Details	Amount
10/03/2009	river view shop	$ 250.00
14/03/2009	psychiatric clinic	$ 270.00
⋮	⋮	⋮

Fig. 8. Credit card record sample

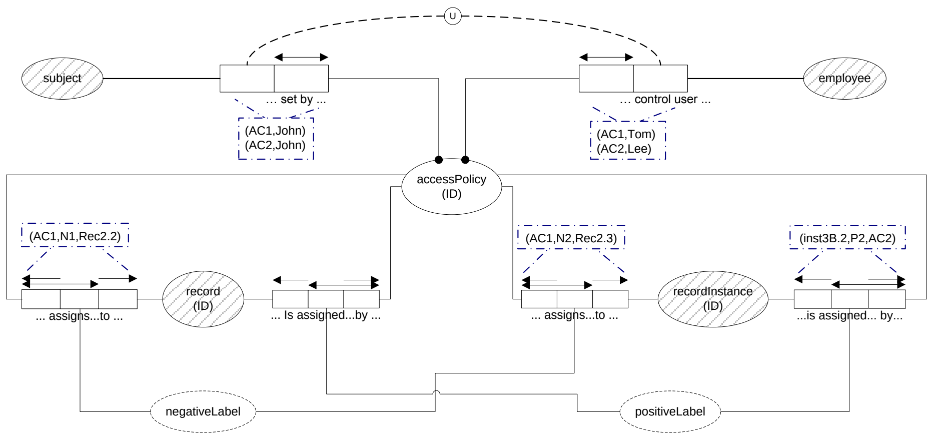


Fig. 9. Conceptual model - authorisations

instance `inst2.1` is related to record `Rec2.1` and captures the account number value '`123423451234321`'.

Figure 9 represents the last part of our conceptual OR model. It shows the authorisation part that captures the subject's privacy requirements. The subject's privacy policy consists of access authorisations that are modelled by entity access policy. Each access policy has a unique ID and must be set by a subject to authorise or restrict the capabilities of certain employees. The access policy can be applied either on the record level, which affects its instances that are owned by the subject, or on a particular record instance. We use positive and negative authorisation approaches to express the subject's required authorisation [8]. In positive authorisation, we use a positive label to flag a certain record or record instance and assign it to an employee. As a result, any employee who has the required positive label of a record or a record instance can access its data, otherwise the employee is disallowed. For example, in Figure 9 `John` has set an access policy `AC2` to authorise only `Lee` to access his record instance `inst3B.2` by adding a flag to the record instance and `Lee`. In negative authorisation, we use negative labels instead of positive labels to restrict certain employees from accessing a certain record or record's instance. For example, in Figure 9 John has set access policy `AC1` to restrict `Tom`'s access to `John`'s credit card form. The access policy `AC1` has two restrictions. One is set on record `Rec2.2` and assigned negative label `N1` and the second is set on record `Rec2.3` and assigned negative label `N2`. The access policy `AC1` implies that `Tom` can not access the data in either record `Rec2.2` or `Rec2.3`.

5 Implementation

To implement these concepts we extended the YAWL environment [14] to accommodate the requirements in Section 3. Our implementation began by converting the conceptual schema introduced in Section 4 to a relational schema. The resulting privacy database tables were then created in the PostgreSQL server. YAWL's Java work-resource allocation framework allowed us to implement a new Java work allocator class that performs the secure allocation strategy. This class interacts with YAWL's and our privacy databases to retrieve information according to the work allocation strategy in Algorithm 1. This algorithm evaluates the restriction weight for each potential participant and then selects the participant who has the lowest restriction weight. In this way, the selected participant is able to access more authorised information than other participants.

To implement our auxiliary data properties (hide and generalise), we took advantage of YAWL's form rendering framework and implemented a new Java class with some additional Java helper classes to receive the form from YAWL's form engine. This class uses the subject ID and participant ID to determine those fields that the participant is not authorised to access, by getting information from our privacy database. The restricted field's attributes are set accordingly to either hide or generalise sensitive data. When the form is returned to the YAWL form rendering engine, it can either totally hide the existence of the restricted field or replace the field's content with generic text.

6 Healthcare Case Study

Here we use a healthcare case study to demonstrate the functionality of our extended workflow engine. For the scenario, we consider a patient's visit to the hospital's emergency department. We modelled the emergency treatment process using the YAWL editor (Figure 10). In this model, various data are retrieved from the patient's Electronic Health Record (EHR) that resides in an external database to be used in the Emergency Room.

Algorithm 1 Least-restricted resource allocation

```
Input: subjectId, taskId
Output: resourceId
Method:
  {:: Find the resources that can execute taskId ::}
  Find the role set RO that can execute task taskId
  Find the resource set RE that can play any role $r \in$ RO
  for all $s \in$ RE do
    s.Weight $\leftarrow 0$
  end for
  {:: Calculate the restriction weight at the record level ::}
  Find the record set RC that is accessed by taskId
  SR $\leftarrow$ number of positive labels that are set in RC by the subjectId
  for all $s \in$ RE do
    SPR $\leftarrow$ number of positive labels that are set by subjectId for s in RC
    SNR $\leftarrow$ negative labels that are set by subjectId for s in RC
    s.Weight $\leftarrow$ $(\mathtt{SR} - \mathtt{SPR}) + \mathtt{SNR}$
  end for
  {:: Calculate the restriction weight at the record instance level ::}
  Find the instance set IN that are accessed by taskId
  SI $\leftarrow$ number of positive labels that are set in IN by the subjectId
  for all $s \in$ RE do
    SIP $\leftarrow$ number of positive labels that are set by subjectId for s in IN
    SIN $\leftarrow$ number of negative labels that are set by subjectId for s in IN
    s.Weight $\leftarrow$ s.Weight $+ (\mathtt{SI} - \mathtt{SIP}) + \mathtt{SIN}$
  end for
  resourceId $\leftarrow$ s, where $\forall x \in$ RE $\bullet$ s.Weight $\leq$ x.Weight
```

The process starts by taking the patient's ID and then a *receptionist* verifies the patient's information to ensure that the patient is the person claimed. Afterwards, a medical preliminary check is carried out for the patient and his current health information (e.g. temperature) are recorded by a *nurse*, followed by a medical diagnosis performed by a *doctor* to determine the patient's status. In this task, the doctor can either proceed and close the patient's case (e.g. by prescribing medications) or request additional information that will be retrieved from the patient's EHR. Once the doctor has accessed the patient's additional information, he can decide to either proceed to close the patient's case or request additional medical support (e.g. a medical consultation, a specific medical service, or both). The doctor, upon receiving the medical report for his request, can either proceed to close the patient's case, request additional medical support, or wait until he receives other requested medical support to do his review. In the model, we defined a cancellation region that is triggered by executing the *close the ER process* task. Once this task is executed, the workflow engine cancels the executing tasks in that region. Finally, the doctor does the necessary check out tasks (e.g. prescribing medication, making an admission request) in the compos-

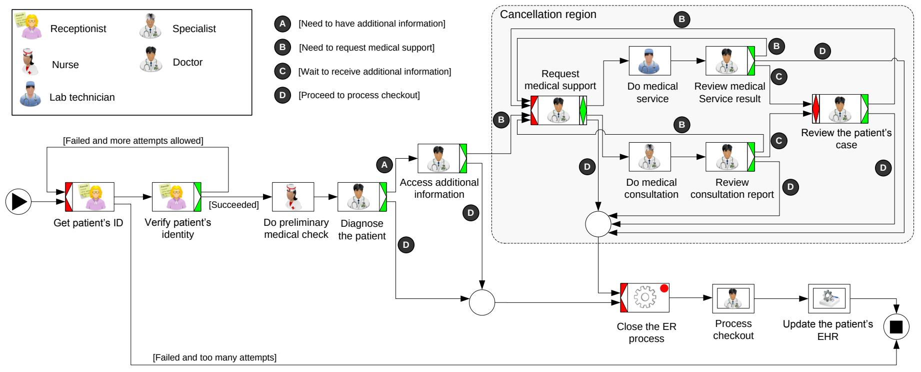

Receptionist
Specialist
Nurse
Doctor
Lab technician
A [Need to have additional information]
B [Need to request medical support]
C [Wait to receive additional information]
D [Proceed to process checkout]
Cancellation region
[Failed and more attempts allowed]
Get patient's ID
Verify patient's identity
[Succeeded]
Do preliminary medical check
Diagnose the patient
Access additional information
Request medical support
Do medical service
Review medical Service result
Review the patient's case
Do medical consultation
Review consultation report
Close the ER process
Process checkout
Update the patient's EHR
[Failed and too many attempts]

Fig. 10. The hospital's emergency process model

(a) Employees and their roles

Employee	Role
Lisa	Receptionist
Jessica	Receptionist
Edith	Nurse
Sara	Nurse
Maria	Nurse
Tom	Doctor
William	Doctor
Sophie	Lab technician
Mark	Specialist

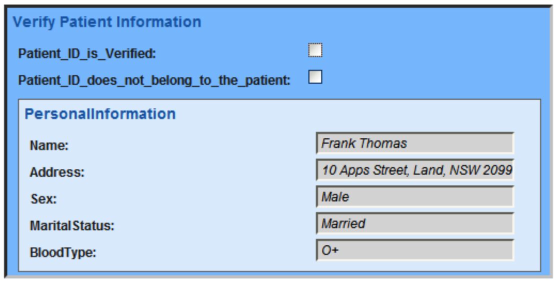
(b) Frank's Access Policy

Authorisation type	Applied to	Assigned to
Negative	DateOfBirth record	Jessica
Negative	DateOfBirth record	Edith
Negative	DateOfBirth record	Sara
Positive	Diagnosis instance 'Chlamydia'	William

Table 1. ER work place setup

ite *process checkout* task, and then the patient's revised medical data produced during this case is uploaded to his EHR in the external database.

In order to execute the emergency room workflow model, we populated our privacy database with data samples. We loaded several patients' EHR data into our database, and set our database tables to reflect the hospital's emergency room employees list and their roles (Table 10(a)). Each role is authorised to execute certain tasks as depicted in Figure 10. As an example, we assumed that a patient Frank has expressed his privacy desire by setting his access control policies as in Table 10(b). In this case, he has denied access to his birth date to

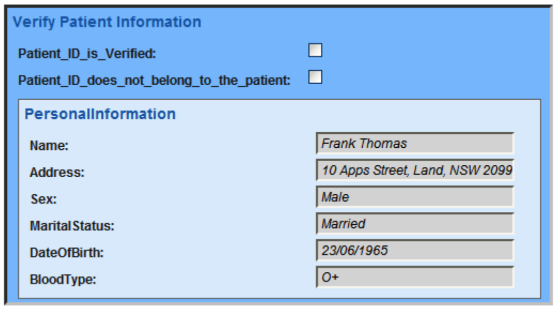

Verify Patient Information

Patient_ID_is_Verified:	☐
Patient_ID_does_not_belong_to_the_patient:	☐

PersonalInformation

Name:	Frank Thomas
Address:	10 Apps Street, Land, NSW 2099
Sex:	Male
MaritalStatus:	Married
BloodType:	O+

(a) Frank's form as presented to Jessica


Verify Patient Information

Patient_ID_is_Verified:	☐
Patient_ID_does_not_belong_to_the_patient:	☐

PersonalInformation

Name:	Frank Thomas
Address:	10 Apps Street, Land, NSW 2099
Sex:	Male
MaritalStatus:	Married
DateOfBirth:	23/06/1965
BloodType:	O+

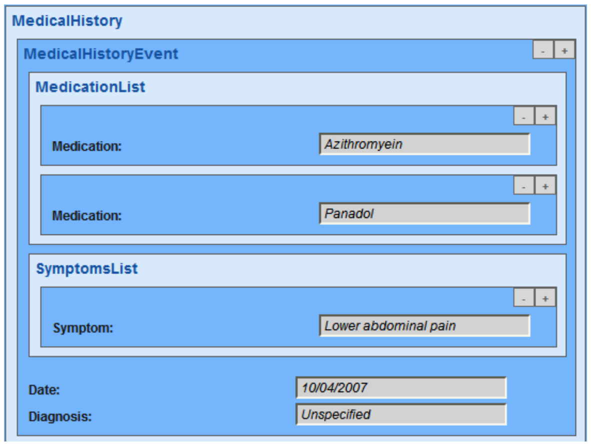
(b) Frank's form as presented to Lisa

Fig. 11. Frank's personal information form

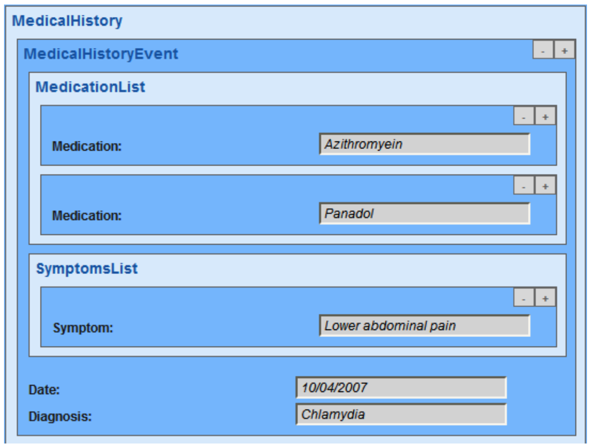
Jessica, Edith or Sara, and has granted access to his Chlamydia diagnosis only to William.

Now let's follow the execution of the hospital's emergency process by assuming that Frank has appeared at the reception desk. Since there is no prior information about the workflow subject, the task *get patient's ID* is assigned randomly to one of the receptionists. If the task is executed by Jessica then she will enter the workflow subject ID, i.e. Frank's ID. Afterwards, she needs to execute they *verify patient's identity* task. The form that is rendered by our extended workflow engine to Jessica is shown in Figure 11(a). The `DateOfBirth` field is entirely hidden by the workflow engine through setting the hide auxiliary property, and is not presented to Jessica because Frank disallows Jessica from seeing this field as per his access policy. By contrast, if the two tasks were assigned to Lisa, the form will show Frank's `DateOfBirth` field as depicted in Figure 11(b) because Frank did not set any restrictions for Lisa.

Once the receptionist has verified Frank's identity, the workflow engine will allocate the task *do preliminary medical check* task to the appropriate nurse. YAWL's workflow engine uses our least restricted-user work allocation strategy (Algorithm 1) to determine the appropriate nurse. In Frank's case, the workflow engine allocates the task to Maria because her restriction weight is the lowest among Sara and Edith (Maria= 0, Sara=Edith= 1).


Medication:	Azithromyein
Medication:	Panadol
Symptom:	Lower abdominal pain
Date:	10/04/2007
Diagnosis:	Unspecified

(a) Frank's form as presented to Tom


Medication:	Azithromyein
Medication:	Panadol
Symptom:	Lower abdominal pain
Date:	10/04/2007
Diagnosis:	Chlamydia

(b) Frank's form as presented to William

Fig. 12. Frank's medical history form

The same allocation strategy is used by the workflow engine to determine a suitable doctor to execute the diagnosis task. This task is allocated by the workflow engine to William because his restriction weight is lower than Tom's. William, as per Frank's privacy policy, is able to access Frank's recorded diagnosis of Chlamydia whereas Tom cannot. The form that is generated to William is depicted in Figure 12(b). However, now let us assume that William is busy and cannot take Frank's case. In this situation, the task will be reallocated to Tom assuming there are no other doctors available. However, the workflow engine knows that Tom is not authorised to know about Frank's Chlamydia diagnosis. Therefore, the workflow engine renders the form so that the Chlamydia diagnosis is replaced by generic term `unspecified` so that Tom does not know about Frank's sensitive diagnosis as shown in the form produced by YAWL in Figure 12(a).

Thus, we have demonstrated how an extended workflow engine is capable of recognising and using the subject of the workflow to best preserve the subject's privacy.

7 Related Work

Authorisation is an important workflow security requirement [6]. It refers to enforcing access control to ensure that only authorised resources are allowed to execute a workflow task. Sandhu introduced a Role-Based Access Control (RBAC) model [24] that breaks the traditional authorisation link between subjects and permissions and inserts a role notion in the middle to ease the authorisation management process. However, the RBAC model is role-centric and does not consider the task notion in WfMSs. Conceptual foundations for Task-Based authorisations are presented by Thomas and Sandhu [23, 25] where privileges for assignment and revocation are discussed in order to provide active access control enforcement. Oh and Park [19] introduced a Task-Role-Based Access Control (T-RBAC) model that is built on top of the RBAC model. A task notion is inserted between roles and permissions, allowing task execution to be assigned to role(s). This new development results in better authorisation management from the workflow perspective. In order to have further access control, authorisation constraints are introduced as additional filters to be applied on subject-role, role-task, and subject-task relations. Bertino et al. [9] presented a formal language for these static and dynamic constraints and provided algorithms to check the inconsistency. However, the T-RBAC model and the authorisation constraints that are introduced do not consider the subject of the workflow in the authorisation policy, hence they fail to address privacy requirements.

Several improvements are introduced to enhance the workflow authorisation and work-resource allocation state. Casati et al. [11] extends the T-RBAC model by adding a new organisational element 'the functional level' and uses event, condition, action rules to present an authorisation constraint. The event part denotes when an authorisation may need to be modified. The condition part verifies that the event actually requires modifications of authorisations, and determines the involved agents, roles, tasks and processes. This authorisation model also fails to address the subject's privacy due to not considering the subject's authorisation policy in its condition part. Access control models that are developed to satisfy the workflow separation of duty security requirement [16–18] consider only the organisation's authorisation policy and thus they too fail to address the workflow subject's privacy rules.

Research has been done to introduce new security constraints to guard workflow execution [5, 30, 31]. However, none has recognised the workflow subject's privacy requirements and therefore neither privacy-constraints nor secure work-resource allocations have been introduced. Cao et al. [10] addressed the importance of human involvement in workflow applications and noted that poor design of work-resource assignment strategies is one of the critical issues in workflow projects. They introduced four different authorisation models, they are department-authorisation, staff-authorisation, role-authorisation, and team-authorisation. These models are utilised to provide a dynamic task authorisation policy. A Task Authorisation Policy Language (TAPL) is introduced to express these dynamic policies. However, this dynamic task authorisation policy fails to address and tackle the workflow subject's privacy concerns because they discuss only the authorisation requirements and did not investigate the workflow subject's privacy concerns. Wolter et al. [28] argue that current process modelling standards are incapable of capturing security goals such as confidentiality, integrity, or dynamic authorisation. Therefore, they proposed a security policy model that contains a set of security constraint models. In the authorisation constraint model, permissions are inserted between a subject (a resource in our case) to a target (a task) but they did not introduce an owner (a subject) for the tasks that are used in the process model. As a result, these authorisation models do not satisfy the privacy requirement for the subject of the workflow. Xu et al. [29] proposed algorithms to optimise resource allocation in order to execute the business process within time and cost constraints. They take into account the structural characteristics of the business process such as task dependencies. However, security constraints have not been considered in their optimised work allocation strategy, so it fails to address not only the privacy requirement but security constraints in general.

jBPM, ruote (OpenWFEru), and Enhydra Shark are open source workflow engines. The jBPM Workflow engine [2] can execute a process written in jPDL and the BPEL process modelling language. jPDL captures process characteristics such as start tasks, simple control flow, and data flow. However, it does not capture the subject of the workflow process which means that the workflow engine is incapable of retrieving the subject's privacy rules in order to integrate them into its simple work-resource allocation model based on users and groups. The ruote workflow engine [3] executes processes that are written in its own process definition language, which comes in two flavours : XML and Ruby DSL. The ruote process definition language fails to introduce the subject of the workflow, as well. Therefore, its work-resource allocation model fails to respect the subject's privacy requirements. The Enhydra Shark workflow engine [1] is also incapable of addressing the privacy problem because its process modelling language (XPDL) does not capture the subject of the workflow.

Commercial workflow management systems are no better than open source WfMSs with respect to satisfying the subject's privacy requirements. The IBM WebSphere WfMS's [15] task allocation algorithm supports direct assignment to resources or indirect assignment via roles. A role is defined by a set of characteristics or by using people assignment criteria that are set at design time, so at run time the workflow engine uses the role's characteristics to define the workflow authorised resources to execute that task. However, the IBM WebSphere WfMS does not consider external constraints in defining its roles (e.g. the subject's privacy rules). In addition, it does not capture the subject's identity in the workflow. As a result, it is incapable of meeting the subject's privacy requirement. TIBCO BPM [4] fails to satisfy the subject's privacy requirement because its work-resource distribution relies only on the defined organisational model and does not consider any external filter in its assignment. FLOWer's [26] work item distribution uses role and resource assignments with some security constraints (e.g. separation of duty). However, a work item's data privacy state is not considered because FLOWer does not consider the identity of the subject of the workflow and thus omits the subject's privacy rules. The COSA [12] BPM defines two access rights for users, they are distribution and authorisation rights. The authorisation rights concern the actions that the user can do on a work item (e.g. re-route, skip, and re-distribute). However, the authorisation rights do not provide a solution to the work item's data privacy requirement. Therefore, the subject's privacy concern over his data used by work items cannot be satisfied because COSA does not consider the workflow subject's identity in its specification and does not provide authorisation at the work item's data level.

8 Conclusion and Future Work

Workflow Management Systems (WfMSs) enforce an organisation's security policy while executing a workflow case to achieve the organisation's security goals. However, they fail to incorporate the security concerns of other entities. Privacy is an important security requirement that pertains to the subject of the data manipulated by the workflow engine. Current WfMSs do not accommodate the subject's privacy policies in their authorisation mechanism.

In this paper, we explained the importance of the privacy requirement and presented its implication for workflow functions. We introduced the workflow subject notion and presented it as part of the workflow specification. This extension allows the WfMS to be aware of the data subject's identity and consequently to retrieve the subject's privacy policy using a workflow data pull pattern. In addition, we presented a new secure work allocation strategy that uses the subject's privacy policy to assign the workflow task to the least-restricted resource from the privacy perspective. Also, we enhanced the workflow form rendering engine's functionality to be aware of private data, using auxiliary data properties, and enforced the appropriate concealment actions. A conceptual OR model was designed to capture these extensions and the whole process was implemented in the YAWL WfMS. We showed through a case study that our extended WfMS is capable of capturing and enforcing a subject's privacy policy.

In future work we will enhance our least-restricted allocation strategy. Currently, our strategy focuses only on the current task and uses the data required by the task in its processing. Therefore, we look to extend it to consider the constraints (e.g. binding and separation of duty) that are set between the current task and other tasks in the workflow case.

References

1. Enhydra Shark: Open source workflow. http://shark.ow2.org/doc/1.1/index.html. [accessed 2009 Aug 20].
2. jBPM user guide. http://docs.jboss.com/jbpm/v4.0/userguide. [accessed 2009 Aug 20].
3. ruote: Open source Ruby workflow engine. http://openwferu.rubyforge.org/documentation.html. [accessed 2009 Aug 20].
4. TIBCO BPM resource center. http://www.tibco.com/solutions/bpm/default.jsp. [accessed 2009 Aug 28].
5. Vijayalakshmi Atluri and Wei-kuang Huang. An authorization model for workflows. In Elisa Bertino, Helmut Kurth, Giancarlo Martella, and Emilio Montolivo, editors, 4^{th} *European Symposium on Research in Computer Security (ESORICS 1996), Rome, Italy, Sept 25-27*, volume 1146 of *Lecture Notes in Computer Science*, pages 44–64. Springer-Verlag, Sept 1996.
6. Vijayalakshmi Atluri and Janice Warner. *Handbook of Database Security: Applications and Trends*, chapter Security for Workflow Systems, pages 213–230. Springer, 2008.
7. Enkh-Amgalan Baatarjav, Ram Dantu, and Santi Phithakkitnukoon. Privacy management for Facebook. In R. Sekar and Arun K. Pujari, editors, 4^{th} *International Conference on Information Systems Security, Hyderabad, India, Dec 16-20*, volume 5352 of *Lecture Notes in Computer Science*, pages 273–286. Springer, 2008.
8. Elisa Bertino, Francesco Buccafurri, and Elena Ferrari. An authorization model and its formal semantics. In G. Goos, J. Hartmanis, and J. van Leeuwen, editors, 5^{th} *European Symposium on Research in Computer Security Louvain-la-Neuve, Belgium, Sept 16–18*, volume 1485 of *Lecture Notes in Computer Science*, pages 127–142. Springer-Verlag, 1998.
9. Elisa Bertino, Elena Ferrari, and Vijay Alturi. The specification and enforcement of authorization constraints in workflow management systems. *ACM Transactions on Information and System Security*, 2(3):65–104, Feb 1999.
10. Jian Cao, Jinjun Chen, Haiyan Zhao, and Minglu Li. A policy-based authorization model for workflow-enabled dynamic process management. *Journal of Network and Computer Applications*, 32(2):412–422, Mar 2009.
11. Fabio Casati, Silvana Casanto, and MariaGrazia Fugini. Managing workflow authorization constraints through active database technology. *Information Systems Frontiers*, 3(3):319–338, 2001.
12. COSA GmbH. COSA BPM 5.7: Process designer manual, June 2008.
13. Terry Haplin. *Information Modeling and Relational Databases: From Conceptual Analysis to Logical Design.* Morgan Kaufmann Publishers, 2001.
14. Arthur H. M. Hofstede, Wil M. P. Aalst, Michael Adams, and Nick Russell, editors. *Modern Business Process Automation: YAWL and its Support Environment.* Springer Berlin Heidelberg, 2010.
15. IBM. WebSphere business modeler, version 6.2.0. http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r2mx/index.jsp?topic=/com.ibm.btools.modeler.advanced.help.doc/doc/concepts/modelelements/processdiagram.html. [accessed 2009 Aug 25].
16. Hao Jiang and Shengye Lu. Access control for workflow environment: The RTFW model. In W. Shen et al., editors, 10^{th} *International Conference Computer Supported Cooperative Work in Design (CSCWD 2006), Nanjing, China, May 3-5*, volume 4402 of *Lecture Notes of Computer Science*, pages 619–626. Springer-Verlag, May 2007.

17. Duen-Ren Liu, Mei-Yu Wu, and Shu-Teng Lee. Role-based authorizations for workflow systems in support of task-based separation of duty. *Journal of Systems and Software*, 73(3):375–387, 2004.
18. Jianxun Liu and Lixia Sun. The application of role-based access control in workflow management systems. In Wil Thissen et al., editors, *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (CSSMC 2004), Hague, Netherland, Oct 10-13*, volume 6, pages 5492–5496. IEEE Computer Society, Oct 2004.
19. Sejong Oh and Seog Park. Task-role-based access control model. *Information Systems*, 28(6):533–562, Sept 2003.
20. Nick Russell, Arthur H.M. ter Hofstede, David Edmond, and Wil M.P. der Aalst. Workflow data patterns: Identification, representation and tool support. In Lois Delcambre, Christian Kop, John Mylopoulos, and Oscar Pastor, editors, 24^{th} *International Conference on Conceptual Modeling, Klagenfurt, Austria, Oct 24-28*, volume 3716 of *Lecture Notes in Computer Science*, pages 353–368. Springer-Verlag, 2005.
21. Nick Russell, Wil M.P. van der Aalst, Arthur H.M. ter Hofstede, and David Edmond. Workflow resource patterns: Identification, representation and tool support. In Oscar Pastor and João Falcão e Cunha, editors, 17^{th} *International Conference on Advanced Information Systems Engineering (CAiSE 2005), Porto, Portugal, June 13-17*, volume 3520 of *Lecture Notes in Computer Science*, pages 216–232. Springer-Verlag, 2005.
22. Pierangela Samarati and Latanya Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, Computer Science Laboratory, SRI International, 1998.
23. Ravi Sandhu and Roshan Thomas. Conceptual foundations for a model of task-based authorizations. In *Proceedings of* 7^{th} *IEEE Computer Security Foundations Workshop (CSFW 1994), Franconia, New Hampshire, USA, June 14-16*, pages 66–79. IEEE Computer Society, June 1994.
24. Ravi S. Sandhu and Pierrangela Samarati. Access control: Principles and practice. *IEEE Communications Magazine*, 32(9):40–48, 1994.
25. Roshan Thomas and Ravi Sandhu. Task-based authorization controls (TBAC): A family of models for active and enterprise-oriented autorization management. In Tsau Young Lin and Shelly Qian, editors, 11^{th} *International Conference on Database Security (DBSec 1997), Lake Tahoe, California, USA, Aug 10-13*, volume 113 of *IFIP Conference Proceedings*, pages 166–181. Chapman & Hall, 1997.
26. Wave-Front. FLOWer 3: Designers guide, 2004.
27. Alan Westin. *Privacy and Freedom.* The Bodley Head Ltd, 1970.
28. Christian Wolter, Michael Menzel, Andreas Schaad, Philip Miseldine, and Christoph Meinel. Model-driven business process security requirements specification. *Journal of Systems Architecture*, 55:211–223, 2009.
29. Jiajie XU, Chengfei Liu, and Xiaohui Zhao. Resource allocation vs. business process improvement: How they impact on each other. In Marlon Dumas, Manfred Reichert, and Ming-Chien Shan, editors, 6^{th} *International Conference on Business Process Management (BPM 2008), Milan, Italy, Sept 2-4*, volume 5240 of *Lecture Notes in Computer Science*, pages 228–243. Springer-Verlag, 2008.
30. Lin Yao, Xiangwei Kong, and Zichuan Xu. A task-role based access control model with multi-constraints. In Jinhwa Kim et al., editors, *Proceedings of* 4^{th} *International Conference on Networked Computing and Advanced Information Management (NCM 2008), Gyeongju, Korea, Sept 2-4*, volume 1, pages 137–143. IEEE Computer Society, Sept 2008.

31. Zhang Yi, Zhang Yong, and Wang Weinong. Modeling and analyzing of workflow authorization management. *Journal of Network and Systems Management*, 12(4):507–535, Dec 2004.