CLEI ELECTRONIC JOURNAL, VOLUME 16, NUMBER 1, PAPER 3, APRIL 2013

Extending the WS-HumanTask Architecture to Support the Resource Perspective of BPEL Processes

Luis J.R. Stroppi
CIDISI, Universidad Tecnológica Nacional, Facultad Regional Santa Fe,
Lavaise 610, S3004EWB, Santa Fe, Argentina,
lstroppi@frsf.utn.edu.ar

and

Pablo D. Villarreal
CIDISI, Universidad Tecnológica Nacional, Facultad Regional Santa Fe,
Lavaise 610, S3004EWB, Santa Fe, Argentina,
pvillarr@frsf.utn.edu.ar

and

Omar Chiotti
INGAR-CONICET,
Avellaneda 3657, S3002GJC, Santa Fe, Argentina,
chiotti@santafe-conicet.gov.ar

Abstract

This work proposes an extension to the WS-HumanTask architecture developed to provide support to the Resource Perspective of BPEL processes. The extension is aimed to overcome limitations identified in the support of BPEL4People and WS-HumanTask to recurrent Resource Perspective requirements described by the Workflow Resource Patterns and well known reference models of workflow management systems. These limitations have been identified through an assessment of BPEL4People and WS-HumanTask against a conceptual model enabling the representation of Resource Perspective requirements by defining three aspects: Resource Structure, Work Distribution and Authorization. The proposed extension specifies an autonomous People Directory component to be integrated with the Task Processor and the Task Client WS-HumanTask components. The behavior of the Task Processor and the Task Client is also extended in order to enable their interactions with the People Directory and to improve their support to the stated Resource Perspective aspects.

Keywords: BPEL4People, WS-HumanTask, Resource Perspective, Workflow.

1 Introduction

A business process defines the tasks to be carried out by the resources of an organization in order to meet a business goal. Business processes involving people are typically supported by information systems with workflow capabilities known as Workflow Management Systems (WfMS) [1]. A WfMS manages the flow of work within an organization through the allocation of tasks to its human resources (resources for short) in the order defined by the process specification. A WfMS can contribute to the efficiency of the process through the elimination of transport and waiting times between tasks by controlling the distribution of work among the organizational resources [2].

Web Services are a widely accepted technology for enterprise application integration. This technology enables the interoperability between applications by using Web standards. WS-BPEL (BPEL) [3] defines a syntax and semantics for describing the behavior of a business process based on the orchestration of Web

Services. BPEL does not take into account aspects related to the participation of people in the execution of business processes. BPEL4People [4] is a BPEL extension based on the WS-HumanTask specification [5] defined with the aim to support a broad range of scenarios that involve people within the execution of business processes. BPEL4People adds workflow capabilities to BPEL by introducing a new type of activity called People Activity which enables the definition of human interactions in processes specifications.

The Resource Perspective of Workflow Management Systems refers to the modeling of resources and their interaction with these systems [6]. A wide range of requirements may arise when integrating people in business processes. These requirements regard the description and grouping of resources, the way in which the work is advertised and bound to the resources for its execution, and the operations they perform to organize and progress the work distributed to them. The Workflow Resource Patterns [7] were developed as an abstraction of a recurring set of such requirements.

In a previous work [8], the Resource Perspective was characterized in terms of three aspects: Resource Structure, Work Distribution and Authorization. An extension to the BPMN 2.0 metamodel and notation [9] was developed in order to enable the definition and visualization of these aspects in process models defined by using BPMN. In a later work [10], a mapping of these extended BPMN models into the syntax elements provided by BPEL4People and WS-HumanTask was also proposed.

The goal of this work is to extend the architecture defined by the WS-HumanTask specification in order to improve its support to the Resource Perspective. The aim is to increase the support provided by BPEL4People and WS-HumanTask to the integration of human beings in business process while keeping the portability of process definitions and the interoperability with preexisting BPEL4People implementations.

This paper is structured as follows. Section 2 presents a conceptual model of the Resource Perspective and introduces concepts of BPEL4People and WS-HumanTask. Section 3 performs an assessment to BPEL4People and WS-HumanTask against the conceptual model presented. Section 4 proposes an extension to the WS-HumanTask architecture to address its shortcomings in the support to the Resource Perspective. Section 5 discusses related work. Finally, section 6 presents the conclusions and future work.

2 Background

This section provides background on the Resource Perspective and concepts of BPEL4People and WS-HumanTask in order to support the analysis and proposals developed in this paper. This section is structured as follows. Section 2.1 identifies three aspects of the Resource Perspective and describes them by means of a conceptual model. Section 2.2 introduces the architecture and the elements provided by BPEL4People and WS-HumanTask for the integration of human beings in business processes.

2.1 Resource Perspective Aspects

Business processes involving people are typically implemented by means of WfMSs. These systems provide support to the Resource Perspective through three components described by well known reference models [1, 11]. The component known as Work Distribution Module [11] performs the allocation of work to the resources based on the process definition. The work allocation may involve the execution of queries to data describing the resources available in the scope of the process or stored in another component called Organizational Repository. Finally, the resources access and complete the tasks distributed to them by means of a software component called Worklist Handler [1].

Figures 1 and 2 depict a conceptual model of the Resource Perspective. The conceptual model represents elements to be defined in specifications interpreted by the aforementioned WfMS components in order to support recurrent Resource Perspective requirements defined by the Workflow Resource Patterns [12]. These elements have been grouped into three aspects: Resource Structure, Work Distribution and Authorization. An extension to the BPMN 2.0 metamodel based on this conceptual model was proposed in previous work [8, 10]. The models presented in these works were adapted to fit the constraints imposed by the BPMN 2.0 extension mechanism.

The Resource Structure aspect is concerned with the definition of information about the resources that is stored in the Organizational Repository. This aspect involves the characterization and classification of resources. The characterization has to do with the description of different features of the resources. The classification is the association of resources with a concept. The classification allows referencing the resources associated with a concept and assigning them common sets of properties. The conceptual model of Figure 1 represents the resource classification by defining *ResourceClassifier* elements and by associating them with zero or more *HumanResource* elements representing individual resources. It also enables defining named relationships between resource classes by means of *ResourceReference* elements and hierarchical relationships between them through *Subsumption* elements. The characterization of resources is enabled by associating zero or more *Parameter* elements with a *HumanResource* or a *ResourceClassifier* element. There are different

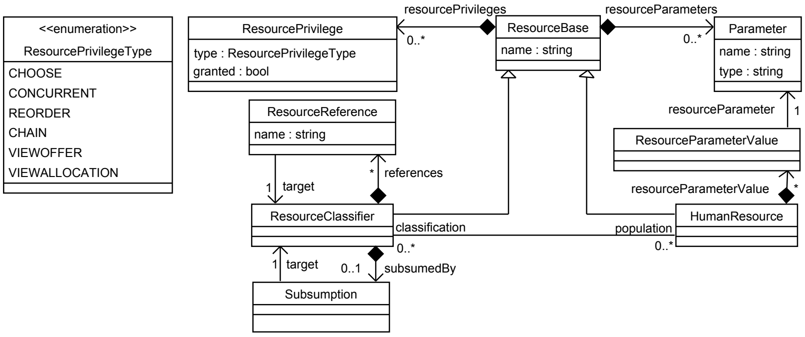


Figure 1: Conceptual Model of Resource Structure.

forms of resource classification [13] such as the functional classification (e.g: roles) or the organizational classification (e.g: organizational units). The *ResourceClassifier* element provides flexibility with regard to the criteria used to classify the human resources of an organization.

The Work Distribution aspect deals with the definition of the behavior of the Work Distribution component of the WfMSs. It defines the way in which the work of a process is advertised and allocated to specific resources for its execution [7]. The work of a process is represented in the conceptual model of Figure 2 by the *Activity* element. An *Activity* represents a piece work that can be composite or atomic. A *UserTask* represents an atomic piece of work (*Task*) to be performed by a human resource. A *UserTask* has an associated *AllocationStrategy* defining how it has to be initially distributed among the resources. It also has zero or more associated *ReallocationStrategy* elements indicating how it has to be redistributed under specific circumstances. These two elements share the most of their structure through their inheritance relationship with *WorkDistributionStrategy*. They define the way the work of a task is distributed to the resources through six properties depicted in Figure 2 by using associations.

The *roleRef* property consists of a reference to a *ResourceRole* element defining a predefined set of task privileges to be granted to the resources to which the task is assigned. The *resourceResolution* property defines the information evaluated to determine the set of resources enabled to perform the specified role in order to progress the work associated with the task. The conceptual model defines two kinds of resource resolution. *ResourceStructureBasedResolution* defines the evaluation of resource structure models defined by using the elements depicted in Figure 1. *DataBasedResolution* specifies the evaluation of data available in the scope of the task. The *resolutionConstraint* property limits the size of the set of resources resulting of the resource resolution process. *SingleResourceConstraint* ensures that the resulting set of resources has a size of one, producing an exception in other case. *SelectedResourceConstraint* specifies a strategy to choose a single resource from the set of resources resulting of the resource resolution process. The conceptual provides support to the *FourEyes*, *RetainFamiliar*, *Random*, *RoundRobin* and *ShortestQueue* identified by the Workflow Resource Patterns [12] (see [8] for more details on them).

The *trigger* property specifies a reference to an event attached to the task being distributed. When defined by *AllocationStategy*, it defers the commencement of the work distribution process. It is mandatory for *ReallocationStrategy* in order to specify the event causing the reallocation of the distributed role. The *distributionAgent* property indicates whether the work distribution is performed by the system (*SystemAgent*) or by a resource (*UserAgent*). The *taskPrivileges* property defines constraints to the set of privileges granted to the resources by the role assigned to them in order to progress the work of the task.

The Authorization aspect refers to the definition of the privileges that the Worklist Handler and the Work Distribution components give to the resources in order to execute the operations they provide. The privileges granted by the Worklist Handler are referred to as Resource Privileges. They define de ability of the resources to execute Worklist Operations to organize the work distributed to them. These privileges depend on the Resource Structure aspect. They are represented in Figure 1 by zero or more *ResourcePrivilege* elements associated with a *HumanResource* or a *ResourceClassifier* element. The privileges granted by the Work Distribution module are called Task Privileges. They depend on the Work Distribution aspect. These privileges define the ability of the resources to execute the so called Work Item Operations in order to progress the work of the task. The Task Privileges are represented in Figure 2 by the *taskPrivileges* property of *WorkDistributionStrategy* defined through its association with *TaskPrivilege*. The set of task privileges granted to a resource as result of the work distribution process is the intersection of the privileges defined by the assigned role and the privileges enabled for the task. The types of resource and task privileges enumerated by *ResourcePrivilegeType* and *TaskPrivilegeType* have been defined based on the requirements specified by the Workflow Resource Patterns [12].

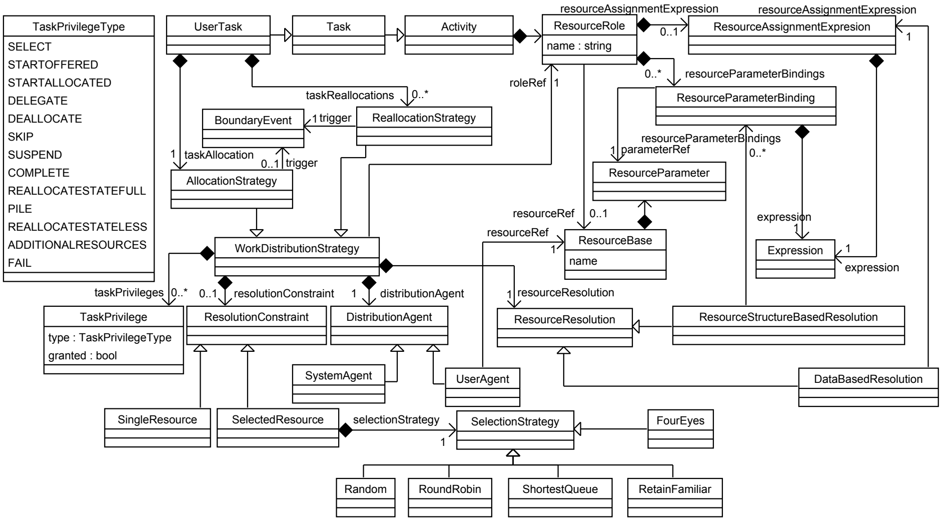


Figure 2: Conceptual Model of Work Distribution.

2.2 The BPEL4People Extension to BPEL

BPEL4People [4] is a BPEL extension that depends on the WS-HumanTask specification [5] for the integration of human beings in the execution of business process. The class diagram of Figure 3 depicts the elements introduced by these specifications to support the Resource Perspective.

BPEL [3] enables the inclusion of new types of activity in process specifications by means of the *ExtensionActivity* element. An *ExtensionActivity* is a wrapper for an element defined in a different namespace that specifies additional features to be supported by the BPEL engine. BPEL4People defines the *PeopleActivity* element that is contained by an *ExtensionActivity* in order to specify a point in a BPEL process in which human resources are required to perform a piece of work.

The *PeopleActivity* element references a *Task* WS-HumanTask element. The *Task* element defines the work that needs to be carried out by people. It contains a *PeopleAssignments* element that specifies the people responsible of performing a generic human role on the task. A WS-HumanTask *GenericHumanRole* defines a set of operations that the resources are enabled to execute in order to handle the work of the task. A *Task* may also define a *startDeadline* specifying the time in which a resource has to execute the *start* operation or a *completionDeadline* specifying the time in which a resource has to execute the *complete* or *fail* operations. It can also specify an *escalation* in order to reassign the people performing the *Potential Owner* generic human role of the task in case that some of the aforementioned deadlines are due.

The BPEL4People specification also extends the BPEL *Process* element with the definition of an element called *HumanInteractions*. This element is the container for *Task* elements described above. It also contains *LogicalPeopleGroup* and *Notification* elements. The *LogicalPeopleGroup* element represents one person, a set of people, or many unresolved groups of people. A logical people group is bound to a people query against a People Directory implementing an Organizational Repository. A *Notification* is a special type of human task that allows the sending of information about noteworthy business events to people. This element is not further discussed as its definition in a process specification does not result in the need for a resource to work.

The following paragraphs introduce the architecture specified by WS-HumanTask in order to manage the life cycle of tasks that need to be performed by people. This architecture is depicted in Figure 4a).

WS-HumanTask defines a Task Processor component that exposes as a service the functionality of the Work Distribution module found in traditional WfMSs. Once a Task Definition specified by using the syntax provided by WS-HumanTask is deployed on a Task Processor, any requestor can create task instances and interact with them. In this architecture, the component that requests the creation of tasks is called Task Parent. The requests for the creation of task instances are performed by invoking an operation of an application-dependent port type referred to as the Task Definition specific interface (Task PT in Figure 4a). In this work, it is assumed that the Task Parent is a BPEL engine supporting BPEL4People. The Task Processor also provides a second interface called the Task Processor's Protocol Handler that provides a coordination framework with the Task Parent in order to handle situations like the prematurely termination or the cancellation of a task.

The Task Processor also handles the People Assignment process. This process determines the resources allowed to claim and work on certain task instance according to the Task Definition. It results in the

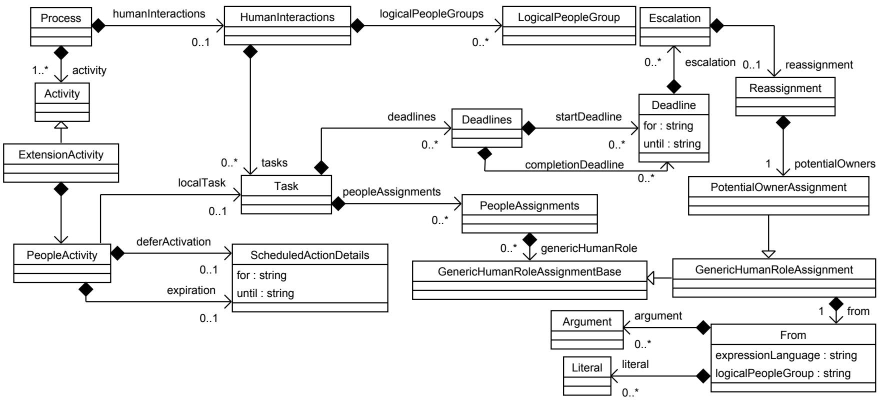


Figure 3: BPEL4People / WS-HumanTask Elements.

assignment of a generic human role to one or more resources in order to perform different operations to change the state of the task instance. The Task Processor usually accomplishes the People Assignment process by executing queries on an People Directory with certain organizational model. The mechanism determining how an implementation evaluates People Assignments and the structure of the data in the People Directory are out of the scope of WS-HumanTask.

The Task Client is the component of the WS-HumanTask architecture aimed to present the task instances to the resources who can claim and work on them. The Task Client provides the functionality of the Worklist Handler module defined in traditional WfMSs. The Task Client enables the end users (resources) to perform operations in order to influence the state of the task instances distributed to them. The execution of these operations results in requests to the Task Client Interface exposed by the Task Processor. The Task Client also provides the user interfaces for the tasks. The details about the creation of user interfaces are out of scope of WS-HumanTask.

3 Support of BPEL4People and WS-HumanTask to the Resource Perspective

This section presents an assessment to the support provided by BPEL4People and WS-HumanTask to the Resource Perspective. The evaluation is performed by establishing correlations between the elements defined by these proposals (see Figure 3) to the Resource Perspective aspects presented in Section 2.1. Table 1 summarizes the support provided to the elements defined in the conceptual model of Figure 2 by the elements depicted in Figure 3 (see also [10]).

The most of the elements defined by BPEL4People and WS-HummanTask provide support to the Work Distribution aspect. The concept of *Generic Human Role* defined by WS-HumanTask has the same semantics than the concept of *Resource Role* defined in the conceptual model of Figure 2. WS-HumanTask also provides support to the concept of *Resource Resolution* by defining *From* elements in *People Assignments.* However, it provides a partial support to the concept of *Resolution Constraints.* It enables defining the *RetainFamiliar* and *FourEyes* selection strategies by using an XPath extension function in a *From* element in order to obtain the Actual Owner of a task instance and assign it to the Potential Owner or Excluded Owner generic human role of another task instance. However, it does not provide support to neither the concept of *SingleResource* resolution constraint nor the *RoundRobin*, *ShortestQueue* or *Random* selection strategies. WS-HumanTask assumes a *SystemAgent* distribution agent when a people assignment for the Potential Owner generic human role is defined. Instead, it assumes the definition of a *UserAgent* distribution agent when this generic human role is left unassigned. WS-HumanTask calls this *Nomination.* The concept of *defer activation* provided by the *PeopleActivity* BPEL4People element is equivalent to the definition of a *trigger* to the *AllocationStrategy* element of a user task depicted in Figure 2. The *trigger* of a *ReallocationStrategy* element is represented in WS-HumanTask by the definition of a start or a completion *Deadline.*

BPEL4People and WS-HumanTask do not support the Resource Structure aspect. Although WS-HumanTask defines the *user* and *group* organizational entities to be returned from the evaluation of people queries, it explicitly excludes from its scope the definition of the structure and interfaces of the People Directory and its interactions with the Task Processor during the People Assignment process.

The Authorization aspect is poorly supported by BPEL4People and WS-HumanTask. WS-HumanTask does not provide support to the concept of *Resource Privilege.* Although it specifies the Task Client Interface exposed by the Task Processor in order to enable its communication with the Task Client, it does not specify

Table 1: BPEL4People/WS-HumanTask support to Work Distribution Strategies.

Work Distribution Concept	BPEL4People/WS-HumanTask Support
Resource Role	Generic Human Role
Resource Resolution	From element of People Assignment
Resolution Constraint	Partially supported by extension functions in from elements
Task Privilege	Not supported.
Distribution Agent	People Assignments or Nomination
AllocationSrtategy Trigger	Defer Activation
RellocationSrtategy Trigger	Escalation

Table 2: Mapping of Task Privileges to WS-HumanTask.

Task Privilege	WS-HumanTask Operation	Generic Human Role
SELECT	claim	Potential Owners
STARTOFFERED	start	Potential Owners
STARTALLOCATED	start	Actual Owner
DELEGATE	delegate	Actual Owner
DEALLOCATE	release	Actual Owner
SKIP	skip	Actual Owner
SUSPEND	suspend	Actual Owner
COMPLETE	complete	Actual Owner
REALLOCATESTATEFULL	forward	Actual Owner
PILE	-	-
REALLOCATESTATELESS	-	-
ADDITIONALRESOURCES	-	-
FAIL	fail	Actual Owner

neither the Worklist Operations to be provided to the resources in order to organize the work distributed to them nor an authorization framework for these operations. WS-HumanTask provides a partial support for *Task Privileges*. It provides a static authorization framework to enforce them based on the assignment of Generic Human Roles to the resources. Table 2 summarizes the mapping between the *Task Privileges* considered in this work, the operations defined by WS-HumanTask and the Generic Human Roles which enable resources to execute them. However, WS-HumanTask does not enable to enforce Task Privileges on a task by task basis.

4 WS-HumanTask Based Architecture to Support the Resource Perspective of Business Processes

This section presents the architecture proposed in this work in order to support the Resource Perspective in BPEL processes. It defines an extension to the architecture specified by WS-HumanTask. The standard WS-HumanTask architecture is extended by introducing a People Directory component exposed as a Web service and by defining its interactions with the Task Processor and the Task Client. This work also proposes extending the features of the Task Processor and the Task Client WS-HumanTask components to overcome the limitations identified in the previous section in their support to the Resource Perspective. The resulting architecture is depicted in Figure 4b. The specification of the services and extensions proposed are available at `http://code.google.com/p/wsht-people-directory/downloads`. This section is structured as follows. Section 4.1 describes the People Directory component proposed to support the Resource Structure aspect. Section 4.2 describes the extensions proposed to the Task Processor to improve its support to the Work Distribution aspect and the Task Privileges. Section 4.3 presents the extensions proposed to the Task Client with the aim to support the Resource Privileges.

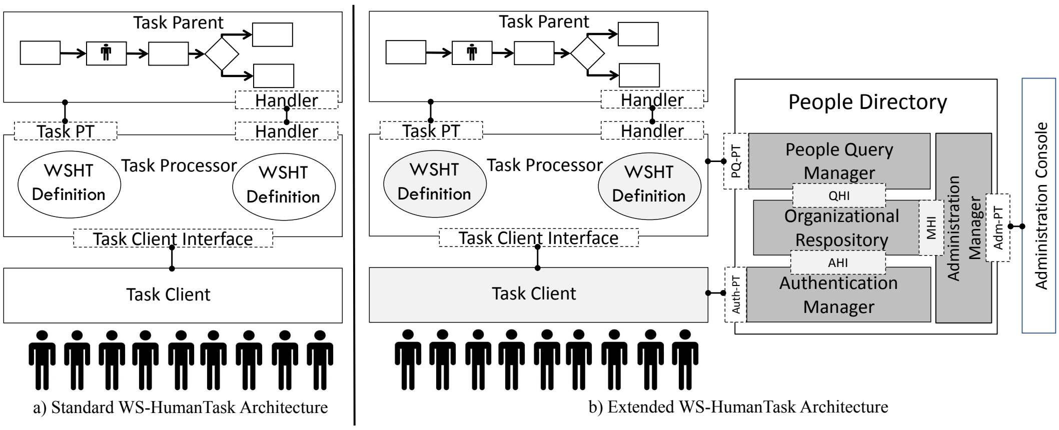


Figure 4: Standard and Extended WS-HumanTask Architecture.

4.1 People Directory

A People Directory component is proposed to provide support to the Resource Structure aspect of the Resource Perspective. This component is exposed as a Web service. It defines the interfaces that enable its integration with the Task Processor and the Task Client by means of WSDL port types. The People Directory consists of three modules designed to provide interoperability with the default WS-HumanTask components and flexibility with regard to the way in which the information about the resources is stored.

The module of the People Directory that provides the means for persisting resource structure models is called Organizational Repository. It enables persisting resource structure models defined in terms of the elements depicted in Figure 1. The Organizational Repository does not impose any restriction with regard to the technology used for persisting resource structure models. Examples of such technologies can be LDAP or a relational database. The details about the underlying technology are isolated to the remaining modules of the People Directory through three interfaces: Query Handler Interface (QHI), Authorization Handler Interface (AHI) and Management Handler Interface (MHI).

The QHI defines the input and output parameters of the methods to handle the execution of people queries against the Organizational Repository. The results of these methods consist of collections of the elements depicted in Figure 1. The AHI specifies the input and output parameters of the methods for validating the identity of the resources against the Organizational Repository. The MHI defines the input and output parameters of the methods enabling the administration of the stored resource structure models.

The People Directory provides a SOAP port type called People Query Port Type (PQ-PT) to enable its integration with the Task Processor. The PQ-PT defines operations for receiving parameterized resource queries from the Task Processor. This port type is implemented by a module called People Query Manager. It consumes the methods defined by the QHI in order to perform the received people queries against the Organizational Repository. The People Query Manager is also responsible for translating the results of the people queries to the data entities defined by the WS-HumanTask specification. The queries resulting in one or more *ResourceClassifier* elements are mapped into *group* WS-HumanTask elements. The queries resulting in one or more *HumanResource* elements are mapped into *user* WS-HumanTask elements.

The People Directory exposes a second SOAP port type called Authentication Port Type (Auth-PT) that provides the interface with the Task Client. This port type enables receiving authentication requests from the Task Client. It is implemented by a module called Authentication Manager. This module consumes the AHI in order to route the received requests to the Organizational Repository. The results of the invocation of the AHI methods are then serialized as outputs of the operations defined by the Authentication Port Type. These results include boolean values indicating whether certain resource has been successfully authenticated or not and the set of *Resource Privileges* granted to that resource.

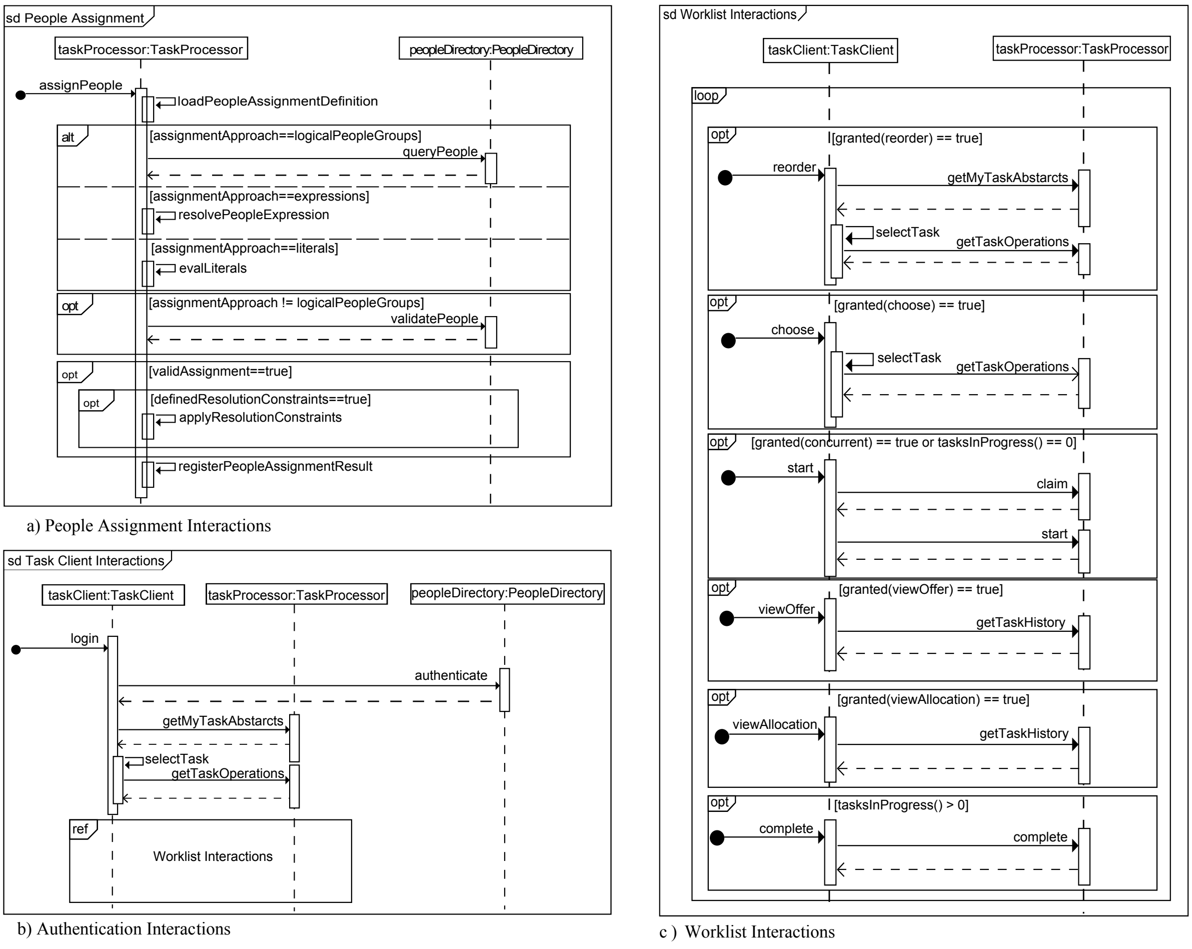
The third SOAP port type defined by the People Directory is called Administration Port Type (Adm-PT). It specifies operations that allow managing and updating the resource structure models stored by the Organizational Repository. Its operations enable the deployment of resource structure models and their redeployment when they are evolved. The Adm-PT is implemented by a module called Administration Manager. This module consumes the methods provided by the MHI in order to manage the stored resource structure models. The Adm-PT enables the People Directory to interact with a separate component called Administration Console. An Administration Console is any application invoking the operations exposed by the Adm-PT in order to enable end users to manage resource structure models.

4.2 Extended Task Processor

The Task Processor is the component specified by WS-HumanTask in order to manage the life cycle of tasks requiring human involvement. WS-HumanTask explicitly excludes from its scope the definition of the mechanism by which the Task Processor queries People Directories. The assessment developed in Section 3 to BPEL4People and WS-HumanTask also revealed deficiencies in their support to the Resource Perspective aspects introduced in Section 2.1. This section specifies the interactions required between the Task Processor and the proposed People Directory for handling people assignment. It also proposes extensions to the capabilities of the Task Processor in order to enforce Task Privileges and Resource Resolution Constraints.

The concept of Resource Resolution introduced in Section 2.1 is supported in WS-HumanTask through the concept of People Assignment. WS-HumanTask provides different ways for determining the resources responsible for acting in a certain generic human role on a task. The following paragraphs describe the interactions that take place between the Task Processor and the proposed People Directory in order to handle the different People Assignment approaches provided by WS-HumanTask. These interactions are depicted in the sequence diagram of Figure 5a.

After receiving a request to start the people assignment process, the Task Processor loads and parses the people assignments defined for the task under consideration. Then, the request continues to be handled according to the people assignment approach specified in the task definition.


sd People Assignment
taskProcessor:TaskProcessor
peopleDirectory:PeopleDirectory
assignPeople
loadPeopleAssignmentDefinition
alt
[assignmentApproach==logicalPeopleGroups]
queryPeople
[assignmentApproach==expressions]
resolvePeopleExpression
[assignmentApproach==literals]
evalLiterals
opt
[assignmentApproach != logicalPeopleGroups]
validatePeople
opt
[validAssignment==true]
opt
[definedResolutionConstraints==true]
applyResolutionConstraints
registerPeopleAssignmentResult
sd Task Client Interactions
taskClient:TaskClient
taskProcessor:TaskProcessor
peopleDirectory:PeopleDirectory
login
authenticate
getMyTaskAbstarcts
selectTask
getTaskOperations
ref
Worklist Interactions
sd Worklist Interactions
taskClient:TaskClient
taskProcessor:TaskProcessor
loop
opt
[granted(reorder) == true]
reorder
getMyTaskAbstarcts
selectTask
getTaskOperations
opt
[granted(choose) == true]
choose
selectTask
getTaskOperations
opt
[granted(concurrent) == true or tasksInProgress() == 0]
start
claim
start
opt
[granted(viewOffer) == true]
viewOffer
getTaskHistory
opt
[granted(viewAllocation) == true]
viewAllocation
getTaskHistory
opt
[tasksInProgress() > 0]
complete
complete

a) People Assignment Interactions

b) Authentication Interactions

c) Worklist Interactions

Figure 5: Component Interactions.

The first people assignment approach consists in using logical people groups. A logical people group represents one person, a set of people or many unresolved groups of people. At deploy time, a logical people group is bound to a people query accepting zero or more parameters against the People Directory. At runtime, the resulting people query is evaluated to retrieve the actual people assigned to the task [5]. The proposed People Directory supports the resolution of logical people groups by means of the *queryPeople* operation exposed by the People Query Port Type. The *queryPeople* operation receives a *from* WS-HumanTask element defining a logical people group and retrieves a set of zero or more WS-HumanTask *user* elements. The *queryPeople* operation supports referencing a resource classifier registered in the People Directory either directly by specifying its name, or indirectly by indicating the name of a resource classifier followed by zero or more references to other resources classifiers separated by dots.

The second WS-HumanTask people assignment approach consists in using expressions. When this approach is taken, the identity of the resources is resolved by evaluating data in the scope of the task. The resolution of a people expression is performed internally by the Task Processor. The third WS-HumanTask people assignment approach consists in using literals. In this case, the identity of the resources is directly specified as part of the task definition. When people is assigned to a task by using the second or the third approach, it is necessary to ensure that the resulting *user* or *group* WS-HumanTask elements represent resources actually registered in the People Directory. The People Query Port Type provides an operation called *validatePeople* for that purpose. This operation receives a set of one or more *user* or *group* WS-HumanTask elements and returns a boolean value indicating whether they correspond to a human resource or a resource classifier registered in the People Directory or not.

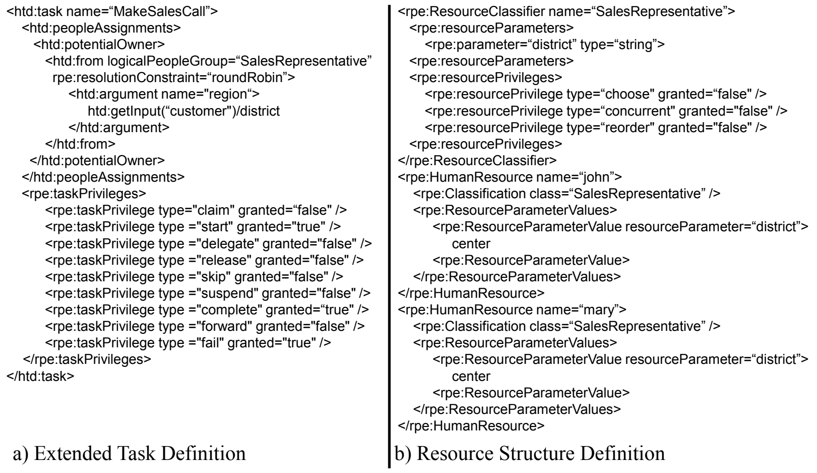
In case that the result of the invocation to *validatePeople* is false, the people assignment process results in an empty set. WS-HumanTask defines the mechanism called *Nomination* in order to handle empty people assignments [5]. In case that the result of *validatePeople* is true, the extended Task Processor continues to apply any resource resolution constraints defined. The details about the application of resolution constraints are discussed below. Finally, the result of the people assignment process is registered in the Worklist handled by the Task Processor.

The assessment performed to BPEL4People and WS-HumanTask in Section 3 also identified deficiencies in the support of these proposals to resolution constraints and the enforcement of task privileges on a task by task basis. The following paragraphs describe an extension to the capabilities of the Task Processor in order to overcome these lacks. This is achieved by extending the syntax provided by WS-HumanTask for the definition of tasks in order to enable the specification of resolution constraints and task privileges. The proposed extensions to the WS-HumanTask syntax are defined by using the extension mechanism provided by the language. Therefore, the portability of the resulting task definitions is not affected. The support to these extensions does not require any change in the predefined interfaces of the Task Processor with the Task Parent and the Task Client. Thus, the interoperability of the extended Task Processor with standard implementations of the Task Parent and the Task Client is also guaranteed.

Two syntactical approaches are proposed for extending the specification of WS-HumanTask people assignments with the definition of resolution constraints. The first approach consists in the use of XPath extension functions surrounding assignment expressions. This approach is already taken by WS-HumanTask in order to support the *RetainFamiliar* and *FourEyes* selection strategies as it was discussed in Section 3. Additional XPath extension functions can be used to support the *RoundRobin*, *ShortestQueue* or *Random* selection strategies. However, this approach can only be applied with people assignments by using expressions. A second approach consists of defining a *resolutionConstraint* extension attribute to the *from* WS-HumanTask element. The advantage of this approach over the first one is that it can be used in conjunction with any of the mechanisms for the definition of people assignments supported by WS-HumanTask. Irrespectively of the approach chosen to define the resolution constraints, they are applied by the Task Processor after the resolution of logical people groups, expressions or literals (see Figure 5a). The People Directory does not have the information required to impose this kind of constraints to the selected resources.

In addition, task definitions are extended with the specification of Task Privileges to be granted or revoked to the resources on a task by task basis. This is achieved through the addition of a *taskPrivileges* element to the *task* WS-HumanTask element. The Task Privileges granted to the resources by Task Processor implementations supporting this extension will be the intersection of the privileges granted by the assigned generic human role and the privileges enabled to be granted in the task definition. At runtime, the proposed extension for task privileges must be supported by modifying the implementation of the *getTaskOperations* operation defined by the Task Client Interface exposed by the Task Processor to the Task Client. This operation must be modified to return the list of operations available to an authorized user given the task privileges specified in the task definition, in addition the role of the user and the state of the task which are considered by default.

An example of extended task definition is depicted in Figure 6a. The task is part of a sales process in which the sales representatives make phone calls to the customers of certain district in order to offer an upgrade to a product they have already bought. The instances of the *MakeSalesCall* task are distributed among the sales representatives in charge of the district in which the customer lives. A *RoundRobin* resolution constraint has been defined to the task by means of the *resolutionConstraint* extension attribute specified to the *from* WS-HumanTask element defining the people assignment of the task. The task definition also includes a detailed specification of the task privileges granted to the sales representatives through the *taskPrivileges* extension element. All the Task Privileges are revoked to the resources with exception of *start*, *complete* and *fail*. The *claim* task privilege does not need to be granted as the outcome of the defined people assignment



```
<htd:task name="MakeSalesCall">
  <htd:peopleAssignments>
    <htd:potentialOwner>
      <htd:from logicalPeopleGroup="SalesRepresentative"
       rpe:resolutionConstraint="roundRobin">
          <htd:argument name="region">
             htd:getInput("customer")/district
          </htd:argument>
      </htd:from>
    </htd:potentialOwner>
  </htd:peopleAssignments>
  <rpe:taskPrivileges>
      <rpe:taskPrivilege type="claim" granted="false" />
      <rpe:taskPrivilege type ="start" granted="true" />
      <rpe:taskPrivilege type ="delegate" granted="false" />
      <rpe:taskPrivilege type ="release" granted="false" />
      <rpe:taskPrivilege type ="skip" granted="false" />
      <rpe:taskPrivilege type ="suspend" granted="false" />
      <rpe:taskPrivilege type ="complete" granted="true" />
      <rpe:taskPrivilege type ="forward" granted="false" />
      <rpe:taskPrivilege type ="fail" granted="true" />
  </rpe:taskPrivileges>
</htd:task>
```

a) Extended Task Definition

```
<rpe:ResourceClassifier name="SalesRepresentative">
  <rpe:resourceParameters>
     <rpe:parameter="district" type="string">
  <rpe:resourceParameters>
  <rpe:resourcePrivileges>
     <rpe:resourcePrivilege type="choose" granted="false" />
     <rpe:resourcePrivilege type="concurrent" granted="false" />
     <rpe:resourcePrivilege type="reorder" granted="false" />
  <rpe:resourcePrivileges>
</rpe:ResourceClassifier>
<rpe:HumanResource name="john">
   <rpe:Classification class="SalesRepresentative" />
   <rpe:ResourceParameterValues>
      <rpe:ResourceParameterValue resourceParameter="district">
         center
      <rpe:ResourceParameterValue>
   </rpe:ResourceParameterValues>
</rpe:HumanResource>
<rpe:HumanResource name="mary">
   <rpe:Classification class="SalesRepresentative" />
   <rpe:ResourceParameterValues>
      <rpe:ResourceParameterValue resourceParameter="district">
         center
      <rpe:ResourceParameterValue>
   </rpe:ResourceParameterValues>
</rpe:HumanResource>
```

b) Resource Structure Definition

Figure 6: Sample Extended Task and Resource Structure Definitions.

consists of a single resource. The specified work distribution strategy provides a means to balance the work load of the sales representatives. The *RoundRobin* resolution constraint results in the assignment of the task instances to them on a cyclic basis. This work distribution strategy is enforced by revoking to the resources all the task privileges enabling them to modify the people assignment performed by the Task Processor. The sales representatives are only able to start working on the task instances and to execute the complete or fail operations to indicate whether they were able to contact the customers or not. The implications of the resource structure defined for this task are discussed at the end of the next section.

4.3 Extended Task Client

The Task Client is the component of the WS-HumanTask architecture that enables people to access and progress the work distributed to them. This work extends the specification of the Task Client component with a description of its interactions with the People Directory proposed in Section 4.1 and the Task Processor. This extension is aimed to overcome the lack of support of BPEL4People and WS-HumanTask to the enforcement of Resource Privileges that has been identified in the assessment presented in Section 3.

The proposed interactions are depicted in Figure 5b and Figure 5c. They start when a resource logs on the Task Client. This makes the Task Client to invoke the *authenticate* operation exposed by the Authentication Port Type implemented by the People Directory. This operation takes a user name and a password as input. It returns an authentication result element consisting of a boolean attribute indicating if the resource has been successfully authenticated and a set of elements specifying the Resource Privileges granted and revoked to the user under consideration.

The interaction protocol continues with the Task Client requesting the list of tasks assigned to the resource under consideration. This is done by invoking the *getMyTaskAbstarcts* operation of the Task Client Interface. The *reorder* Resource Privilege grants to the resource the ability of modifying the value of the *order by* input parameter of *getMyTaskAbstracts*. The default value of this parameter is by priority.

When the resource selects a task to work on it, the value of the *choose* Resource Privilege gives to the resource the ability to pick a task different from the first shown in the list. Then, at the moment of initiating the execution of the task, the *concurrent* Resource Privilege determines whether the resource is enabled to have more than one task assigned in the state *in progress*.

The protocol finishes with the resources completing the execution of the tasks allocated to them. At any point in the interaction protocol the resources can query the history of the tasks assigned to them. This is supported by the *getTaskHistory* operation provided by the WS-HumanTask Task Client Interface. The *viewOffer* and *viewAllocation* Resource Privileges limit the ability of the resources to know the set of resources assigned to the task or who have claimed the task in order to complete it.

The *chain* Resource Privilege cannot be granted by this architecture. This is because WS-HumanTask task definitions are agnostic with regard to the order in which the tasks take place within the process. These details are defined by the elements provided by BPEL4People. Such details are not available to the Task Client through the interfaces exposed by the Task Processor.

The resource structure definition depicted in Figure 6b revokes to the sales representatives the *choose*, *reorder* and *concurrent* Resource Privileges. It has been designed in order to enforce the work distribution strategy described in the previous section. The sales representatives are forced by the Task Client to complete the instances of the *MakeSalesCall* task in the order assigned by the Task Processor. They are also prevented to start working on more than one task instance at the same time.

5 Related Work

There exists previous work on the integration of people in the execution of processes supported by BPEL4People and WS-HumanTask.

A security framework for WS-HumanTask Task Processors was proposed in [14]. The framework includes a security token service that provides support to the pattern of brokered authentication and authorization of the Kerberos authentication scheme. Both the authentication of actors and their authorization to perform work item operations are delegated to the framework outside of the Task Processor by using a token-based approach. This requires the addition of extra components in order to handle the revocation of tokens in case of cancellations of tasks or due deadlines. That proposal provides an authorization framework only for work item operations. Our proposal defines an authorization framework for both work item and worklist operations and keeps a simpler architecture by leaving the enforcement of Task Privileges to the Task Processor and the enforcement of Resource Privileges to the Task Client.

Another work enabling the integration of human beings in BPEL processes was presented in [15]. It provides an approach based on the definition of an XSL template in order to transform BPEL4People process specifications into standard BPEL. People activities are transformed into invoke and receive activities. The proposal defines a People Activity Manager component that replaces the Task Processor specified by WS-HumanTask. That component provides a static role-based authorization framework for work item operations as the WS-HumanTask one does. However, the set of work item operations provided by the People Activity Manager is smaller than the specified by WS-HumanTask.

VieBOP [16] is another proposal that enables handling BPEL4People processes reusing legacy BPEL engines. It defines a complex deployment process in which people activities are transformed into invoke and receive BPEL activities in order to delegate the management of human tasks to a component provided by the platform. The extended architecture proposed in the present work keeps the standard interfaces defined by WS-HumanTask for the Task Parent, the Task Processor and the Task Client. This enables the interaction with any BPEL4People engine. Also, the reuse of legacy BPEL engines can still be supported through an approach similar to the proposed in [15] based on XSL transformations.

6 Conclusions

This work proposed an extension to the architecture defined by WS-HumanTask with the aim to improve the support provided by BPEL4People and WS-HumanTask to the Resource Perspective. This extension is based on a conceptual model of this perspective. This conceptual model incorporates common characteristics of well known reference models and architectures of WFMSs. The elements of this conceptual model are grouped into three aspects: Resource Structure, Work Distribution and Authorization. The need for extending the WS-HumanTask architecture has been determined through the assessment performed to the support provided by BPEL4People and WS-HumanTask to these aspects.

An autonomous People Directory has been proposed in order to address the Resource Structure aspect, which is not covered by the WS-HumanTask specification. The People Directory allows storing and querying organizational models defining the characterization and classification of resources. The People Directory provides flexibility with regard to the technology used for persisting the resource structure models. The features of the People Directory are exposed as a service through the People Query, Authorization and Administration port types. The interactions required to enable the Task Processor to perform queries against the proposed People Directory during the People Assignment process have also been defined.

This work has also proposed extending the Task Processor component specified by WS-HumanTask in order to improve its support to the Work Distribution and Authorization aspects. The extension enables the Task Processor to support resource resolution constraints and task privileges. Two syntactical approaches have been proposed in order to extend the definition of tasks with resolution constraints. An additional one has been provided for the definition of task privileges. These approaches can be applied by using the extension mechanism supported by WS-HumanTask. Therefore, the portability of the resulting task definitions is kept. The additional behavior to be supported by the Task Processor does not require modifying its interfaces with the Task Parent and the Task Client. Therefore, the interoperability of the extended Task Processors with Task Parent and Task Client implementations compliant with the WS-HumanTask specification is also guaranteed.

Finally, an extension has been proposed for the Task Client in order to improve its support to the Authorization aspect through the enforcement of Resource Privileges. The extension consists of the addition to the Task Client of the ability to obtain from the People Directory the privileges granted to the resources in order to organize the work distributed to them. This is achieved by invoking the operations defined by the Authorization port type exposed by the People Directory.

The implementation of the components and extensions defined in this work in order to support the Resource Perspective of business processes and the definition of a model driven development method to generate executable specifications of the Resource Perspective from business process models defined at high level of abstraction are considered as future work.

References

[1] D. Hollingsworth. (1995, January) The workflow reference model. Workflow Management Coalition.

[2] M. Zur Muehlen, "Organizational Management in Workflow Applications - Issues and Perspectives," *Information Technology and Management*, vol. 5, pp. 271–291, 2004, 10.1023/B:ITEM.0000031582.55219.2b.

[3] OASIS, "Web Services Business Process Execution Language for Web Services, Version 2.0," April 2007. [Online]. Available: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

[4] M. Kloppmannd, D. Koenig, F. Leymann, G. Pfau, A. Rickayze, C. von Riege, P. Schmidt, and I. Trickovic, "WS-BPEL Extension for People - BPEL4People," july 2005.

[5] A. Agrawal, M. Amend, M. Das, M. Ford, M. Keller, C.and Kloppmann, D. König, F. Leymann, R. Müller, G. Pfau, K. Plösser, R. Rangaswamy, A. Rickayzen, M. Rowley, P. Schmidt, I. Trickovic, A. Yiu, and M. Zeller, "Web Services HumanTask (WS-HumanTask), Version 1.1," August 2010.

[6] *Process-aware information systems: Bridging people and software through process technology.* Journal of the American Society for Information Science and Technology, Vol. 58, No.3, pp. 455-456, 2007.

[7] N. Russell, A. H. M. T. Hofstede, and D. Edmond, "Workflow resource patterns," 2004.

[8] L. J. R. Stroppi, O. Chiotti, and P. D. Villarreal, "A BPMN 2.0 Extension to Define the Resource Perspective of Business Process Models," in *CIbSE 2011: Proceedings of the XIV Iberoamerican Conference on Software Engineering*, Apr. 2011.

[9] OMG, "Business Process Model and Notation (BPMN), V.2.0," http://www.omg.org/spec/BPMN/2.0, 2011.

[10] L. J. R. Stroppi, O. Chiotti, and P. D. Villarreal, "Extended Resource Perspective Support for BPMN and BPEL," in *CIbSE 2012: Proceedings of the XV Iberoamerican Conference on Software Engineering*, Apr. 2012.

[11] M. Pesic and W. van der Aalst, "Modeling Work Distribution Mechanisms Using Colored Petri Nets," 2005.

[12] N. Russell, W. M. P. van der Aalst, A. H. M. ter Hofstede, and D. Edmond, *Workflow Resource Patterns: Identification, Representation and Tool Support*, ser. Lecture Notes in Computer Science. Heidelberg: Springer Berlin, 2005, vol. 3520, ch. 16, pp. 216–232–232.

[13] W. van der Aalst and K. van Hee, *Workflow Management: Models, Methods, and Systems (Cooperative Information Systems).* The MIT Press, Mar. 2004.

[14] M. Gerhards, S. Skorupa, V. Sander, P. Pfeiffer, and A. Belloum, "Towards a Security Framework for a WS-HumanTask processor," in *CNSM'11*, 2011, pp. 1–5.

[15] J. Thomas, F. Paci, E. Bertino, and P. Eugster, "User Tasks and Access Control over Web Services," in *Proceedings of the 15th IEEE International Conference on Web Services*, 2007.

[16] T. Holmes, M. Vasko, and S. Dustdar, "VieBOP: Extending BPEL Engines with BPEL4People," in *16TH EUROMICRO INTERNATIONAL CONFERENCE ON PARALLEL, DISTRIBUTED AND NETWORK-BASED PROCESSING.* IEEE Computer Society, 2008, pp. 547–555.