Master's Thesis

BPEL4People
Extending WS-BPEL for People

carried out at the
Information Systems Institute
Distributed Systems Group
Vienna University of Technology

under the guidance of
o.Univ.Prof. Mag. Dr. Schahram Dustdar

contributing assistant
Univ.Ass. DI Martin Vasko

by

Ta'id Holmes[1]

Vienna, 20. March 2007

[1] © 2007 Ta'id Holmes
tholmes@iaeste.or.at
http://taid.holmes.at
Matr.Nr. 9625914

Contents

Abstract

BPEL, an XML based language, formally describes business processes and business interaction protocols. While WS-BPEL processes support automated process integration, there is a need to support human tasks, which naturally comes with new requirements. BPEL4People is a joint project of IBM and SAP that describes scenarios where users are involved in business processes, and defines appropriate extensions to WS-BPEL, exclusively using web service interfaces for maximum interoperability. Integration of human tasks into BPEL processes is achieved by defining a people activity, that encapsulates a human task, as a concrete implementation of a BPEL activity. This thesis proposes a concrete specification of BPEL4People that defines syntax and semantics that complies with the WS-BPEL specification. Finally, a generic BPEL4People system is realised that can to be coupled with a BPEL engine in order to integrate human tasks into business processes.

List of Figures

List of Tables

Chapter 1

Introduction

Living in a competitive world, businesses are naturally interested in information technology supporting them for competitive advantage [23]. As cooperation becomes more and more important for companies in order to stick out of the market [12], new challenges arise for the support of business to business scenarios by information technology [29].

While enterprises already can profit using traditional workflow management systems [15], BPEL[1], a business process execution language that permits formal specification of business processes, enables companies to collaborate with each others using modern information technology [47].

Formalisation as a matter of fact brings several benefits with it:

- By formalising a business process it becomes well reflected: problems can be identified at an early stage so that the whole process can be optimised opportunely.

- The formalisation itself represents a kind of contract: misunderstandings and disputes can be reduced significantly.

- Process instances can be monitored, compared and analysed: costs can be localised and quantified.

- Finally certain process activities may be outsourced enabling the company to concentrate on key competences while collaborating with business partners.

[1] see also 5.1.4

BPEL can be ideal for automated processes between businesses using respective services. However the obvious scenario of a business process that depends on a person to fulfil a certain task as a kind of process activity, is not yet covered by BPEL.

In this chapter we will define the problem, enumerate respective requirements for BPEL4People and look at related work.

Chapter 2 will describe the results of this thesis. First the requirements mentioned in the following section will be elaborated. The syntax and semantic of BPEL4People – an extension for BPEL – will be presented that enables the latter to embrace scenarios where humans participate in processes. Also the architecture of a generic BPEL4People system, enabling traditional BPEL engines to handle BPEL4People processes transparently, will be introduced.

Chapter 3 will discuss the results. Again we will first start with the basic requirements before surveying other results.

Chapter 4 will give a conclusion by also referring to further work.

Chapter 5 We will then give a theoretical background introducing underlying technologies.

1.1 Problem Definition and Motivation

Web services have become widely accepted as the de facto standard for distributed business applications [2]. They bring maximum interoperability and use an open and flexible architecture. The implementation and complexity of a web service can be hidden towards a caller. Moreover BPEL, that formally describes processes, permits orchestration of web services [9].

While external activities within BPEL correspond to web services, the obvious business activity of a human interaction within a process is not covered by BPEL [1].

This human interaction, for example, might be to give an approval at a certain stage of the process.

How could a simple interaction be realized with BPEL?

As a matter of fact, the approval activity has to be designed within the BPEL process as an external activity to invoke. This web service then has to be individually implemented and besides receiving the web service call from the

BPEL process and returning it as a result, it has to notify a user and offer a web interface for giving the approval accordingly.

Thus: while BPEL glues together the logic of a process, individual solutions have to be realised when people are integrated into business processes. One may argue that it is justified to accept this circumstance as a customised and optimised user interface for end users makes sense.

What to do however if this approval must only be given by someone part from a certain group of people? In this case, one would be quick to suggest, a database containing data on human resource information has to be consulted. It becomes complicated from a developers point of view if an approval cannot be given immediately but requires further investigation. In this case the person, if authorized, might undertake the task of performing related work. Our implementation then has to document that of course. While on the one hand we are interested not to give the same task to different people, we want users with an administrative role to observe and interact with the process nonetheless.

Actually, these few considerations already bring us close to BPEL4People as BPEL4People focuses on integrating people into processes. A company may be structured in a static and hierarchical way and it may very well have business processes that fit into a system of defined roles. But as we can see this places lots of restrictions on a far too limited system.

What we want in fact is a formal description of how people interact with processes. Each process may be disparate from others and the manning of human roles may differ completely. Most interestingly: people assignment does not have to take place within the same enterprise. This is to say employers of several companies may be delegated to work within various inter-corporate groups occupying different roles respectively.

Now how would one model and realise all that without proper formal specification on how people interact with processes?

As we have seen there is a need to integrate people into BPEL processes.

IBM and SAP have recognised the importance of extending BPEL for people and published a white paper [1], that addresses various BPEL4People requirements and scenarios[2].

While supporting the idea of decentralisation and transparency in regard

[2] see also 1.2.1

to process activities by means of interoperability, BPEL has to be enriched by an additional group of syntax and semantics, that focus on the description of how people interact with a process.

In order to gain an understanding of the new requirements that arise from the formal integration of people into processes, we first want to analyze human-process interactions.

1.1.1 Human ⟷ Process Interaction

Interaction between a given process and a human and/or vice–verso can be categorised into the following basic types:

Human → Process

Unidirectional interaction initiated by a human can be categorised as following:

instantiation The most simple interaction of this type. A user instantiates a process.

supply data Data can be supplied by a user for the process. This can be the result of a performed task as we will see as well as any annotation that may be interesting information for the continuation of the process. This may take place as early as when instantiating the process.

Human ⇌ Process

Bidirectional interaction between a process and a human consists of:

request data A user is notified by a process and further data is required for the process execution to proceed. Upon notification the user typically will perform a task and submit the necessary data to the process.

provide data A task is performed by a user after having been notified and the results is sent back to the process.

> **approval** Approval, in fact, can be understood as the most simple case of a user providing data: a binary decision. Nonetheless an approval can be enriched by an additional annotation that may document the decision.

Human ← Process

Notification is the only unidirectional interaction that can be initiated by a process:

notification The most simple interaction of this type. A user is notified by the process that continues its work flow. While notification does not block the process it can be pointed out that the interaction type where data may be provided, does.

1.1.2 Role-based Interactions of People

In everyday life we are used to play several roles when interacting with our environment. For this we do not require hierarchical structures with explicitly defined roles. It can be observed in particular that responsibilities are distributed between parties, during an interaction.

1.1.3 People Activity: a Human Task

The execution of a task that represents a small unit of work and that can be accomplished in an atomic way is delegated to a person only. A task may be equal to another task that is assigned to another user[3] but the same task must not – and by definition actually cannot – be performed twice.

1.1.4 Scenario

Scenario that must be covered by BPEL4People as additional requirements as mentioned by [1] are:

four eyes scenario A task may be given to persons that are independent in order to obtain different opinions or for comparison of results. In such a scenario, although the set of potential owners might be the same, we do not want the owner of one task to also claim another identical task. If a potential owner claims one of the equal tasks they must be excluded as potential owner for all others.
This scenario requires thus that a potential owner might only claim one out of multiple equal tasks. Also the information of the actual owners must not be disclosed by the system in order to avoid any collusion between them.

[3] see the definition for the four-eyes principle 1.1.4 for an example

nomination A potential owner may be nominated to become the actual owner of a task. In such a scenario even though there might be other potential owners, only nominated owners shall be allowed to claim a task.

escalation An escalation takes place if deadlines are not met. In such a case escalation notifications to escalation recipients take place.

chained execution Tasks may be logically chained - that is to say a task might link to a following, related task. Chained execution is an execution of a process fragment where the owner of one task by successfully completing a task, automatically requests ownership for the following chained task.

1.1.5 Goals

We want to summarise the goals for BPEL4People:

Within the context of a business process BPEL4People must

- support role based interaction of people
- provide means of assigning users to generic human roles
- take care to delegate ownership of a task to a person only
- support scenario as mentioned above

by extending BPEL with additional independent syntax and semantic.

1.2 Related work

1.2.1 BPEL4People White Paper

IBM and SAP have published a white paper [1] that describes requirements for BPEL4People. We have analysed[4] and taken this as a basis for this thesis.

1.2.2 Alternatives

While the formal integration of humans into modern business process languages as BPEL is still a virgin topic, process modelling has evolved over years. As a matter of fact, plenty of workflow languages exist nowadays.

[4] see also chapter A

- **Petri nets** represent an abstract way of visualising processes [18] that BPEL can be transformed to [26]. BPEL itself evolved out of
- Web Services Flow Language (WSFL) [19] and
- XLang [39]. [27] compares BPEL with the
- **Business Process Modeling Notation** (BPMN) [34] and with
- **Yet Another Workflow Language** (YAWL). Another process description language is the
- **XML Process Definition Language** (XPDL) [46]. It supports complete graphical representation of BPMN. BPMN itself can be transformed to BPEL.

As model driven architectures [33] become more and more important, the application on processes represents an interesting undertaking. [36] discusses the suitability of UML activity diagrams for business process modelling. A framework for generating XPDL specifications from UML activity diagrams finally is presented by [17].

Chapter 2

Results

- For role-bases interaction actors and use cases have been identified (2.1).

- Human tasks are protected as critical resources from potential owners (2.3).

- Scenario are modelled using state charts and activity diagrams (2.4).

- A proposal defining BPEL4People syntax and semantics has been carried out (2.5).

- A reference implementation for a BPEL4People system that can be taken as a basis for being coupled with any BPEL compliant engine has been developed (2.6).

2.1 visualising Human-Process Interaction

Sequence diagrams for human-process interaction are found in the following tabular use cases corresponding figures.

use case	figure
instantiation	2.1
notification	2.2
people activity	2.3

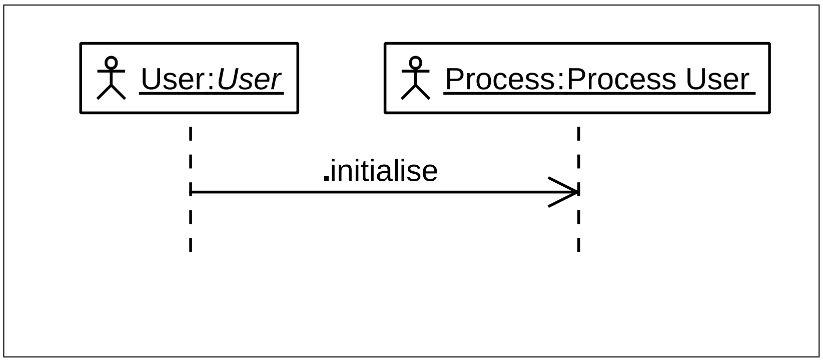

Figure 2.1: Sequence Diagram for Instantiation

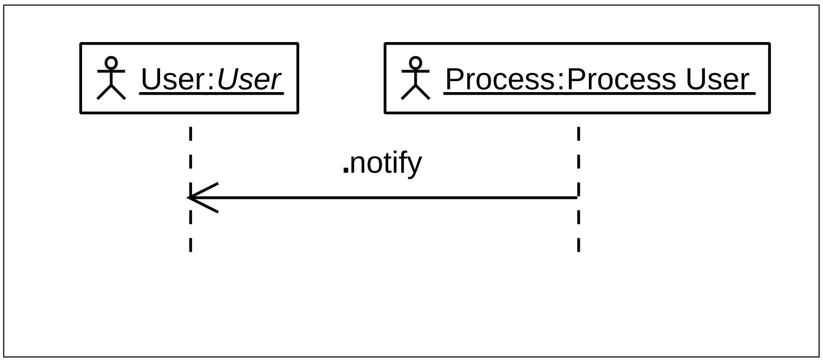

Figure 2.2: Sequence Diagram for Notification

2.2 describing Role-based Interaction

2.2.1 Actors

Business Administrator A business administrator is interested in all process instances of a specific process class. Figure 2.4 shows a use case diagram for the business administrator.

Escalation Recipient A person that receives a notification if specified deadlines have not been met. Figure 2.5 shows a use case diagram for the escalation recipient.

Owner A potential owner that successfully claimed a people activity. Figure 2.6 shows a use case diagram for the owner.

Potential Owner A person that may claim and complete a people activity. Figure 2.7 shows a use case diagram for the potential owner.

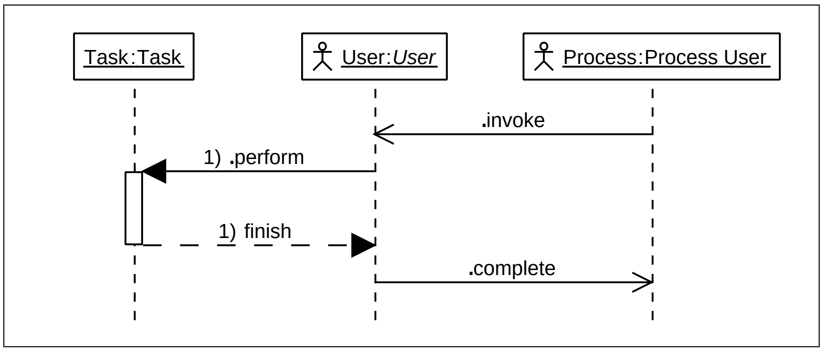

Figure 2.3: Sequence Diagram for People Activity

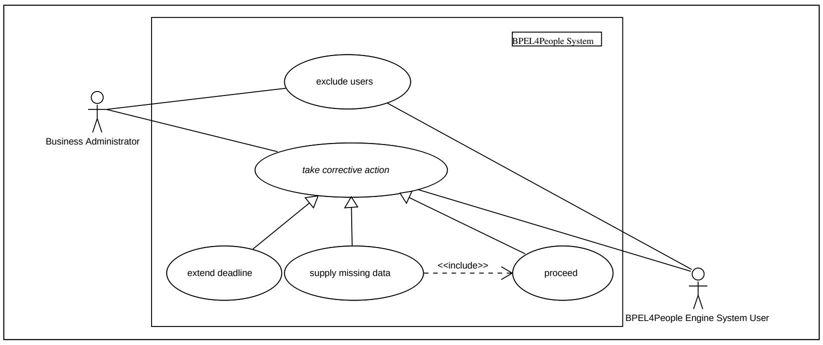

Figure 2.4: Use Case Diagram for a Business Administrator

Nominated Owner A potential owner that has been nominated for ownership. Figure 2.7 shows the nominated owner as a specialisation of the owner.

Process Initiator A person that may create an instance of a process or in case of a process instance the person who actually initiated the process. Figure 2.8 shows a use case diagram for the process initiator.

Process Stakeholder A person that can observe and influence a process instance. Figure 2.9 shows a use case diagram for the process stakeholder.

Supervisor Ownership nomination can be performed by a supervisor that can watch a process instance and access corresponding data. Figure 2.10 shows a use case diagram for the supervisor.

BPEL4People Administrator A person that defines new processes and is able to assign roles. Figure 2.11 shows a use case diagram for the

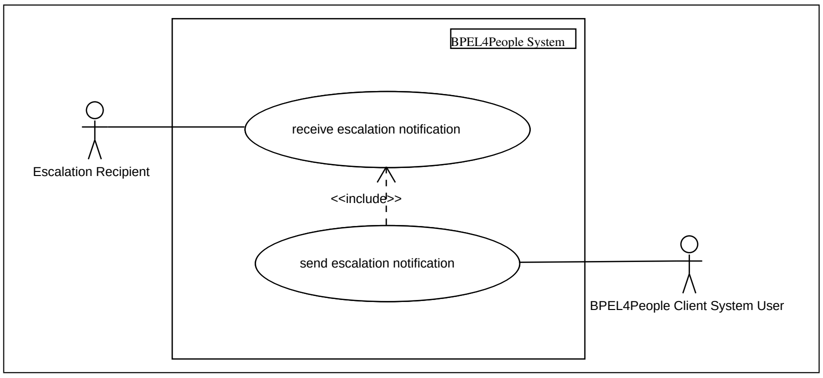
BPEL4People System
receive escalation notification
Escalation Recipient
<<include>>
send escalation notification
BPEL4People Client System User

Figure 2.5: Use Case Diagram for an Escalation Recipient

BPEL4People administrator. As business objects are generic abstract objects (relate to figure 2.12) the BPEL4People administrator has full control over BPEL4People specific data. Some important and typical use cases that are exercised by BPEL4People administrators are shown in Figure 2.13

2.2.2 Use Cases

assign people Roles have to be assigned to people before, while or after process initialisation. A Business Administrator for instance might want to nominate someone for a task. Thus nomination makes use of this use case.

supply ad hoc attachment Ad hoc attachments may be supplied during a process by knowledge-workers. These attachments are visible for the process and may for instance document a decision.

supply data In order to complete a task, some data may be necessary to be supplied.

notify for new work item Nominated owners should be notified of their nomination. Such a notification might make sense for potential owners in general as well.

claim task A potential owner can claim a task if it's state is ready and not nominated.

complete task A task can be completed successfully or unsuccessfully by the task's owner or a business administrator.

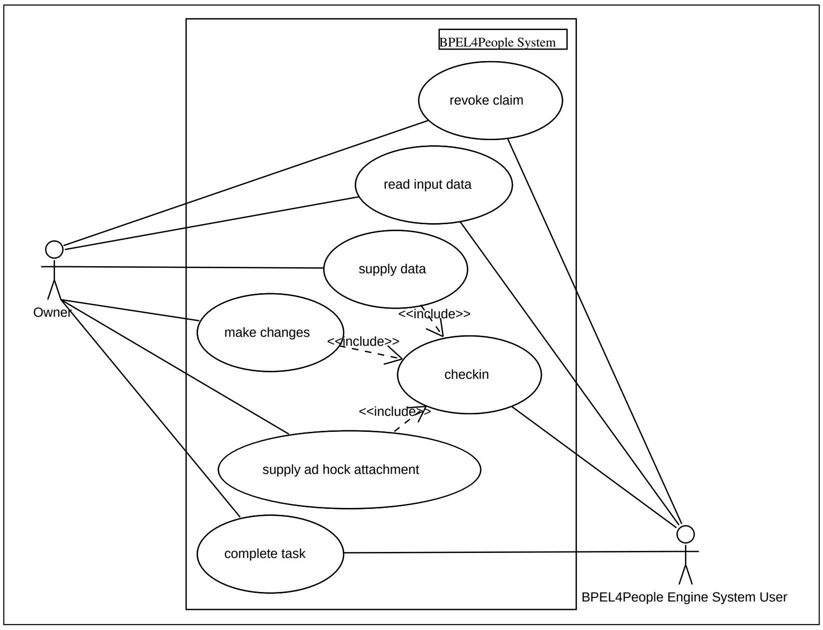

Figure 2.6: Use Case Diagram for a Owner

make changes Work may be submitted without altering an owned tasks state. Revision control is then a feature that a BPEL4People system should provide.

read input data A tasks input data that is relevant for accomplishment.

receive escalation notification An escalation notification is received by an escalation recipient. Escalation notifications are sent by the BPEL4People system if deadlines time out.

decide on corrective action A business administrator may decide on a corrective action for an escalated task. He might complete the task by supplying missing data himself or by extending deadlines. Also he can decide to let the process continue without the successful termination of the task when appropriate.

exclude users The feature of being able to exclude users explicitly is mentioned here as this is an important requirement for a BPEL4People system although it is covered by the assign people use case implicitly that must be capable of not only assigning a group of people or individuals to a role but also to exclude them set theoretically from such an assignment.

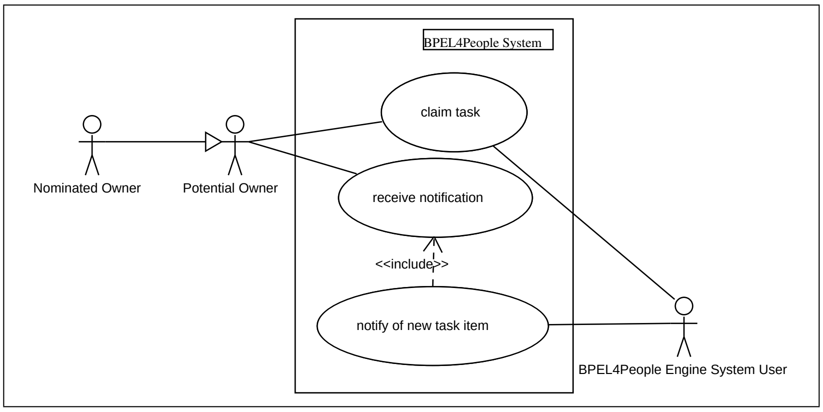

Figure 2.7: Use Case Diagram for a Potential Owner

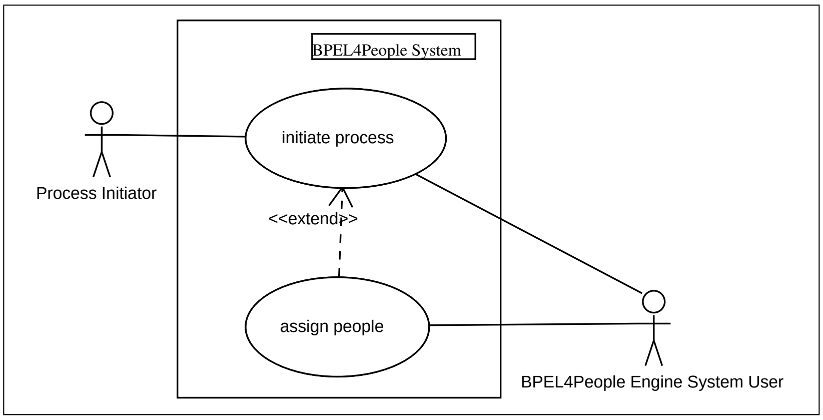

Figure 2.8: Use Case Diagram for a Process Initiator

2.3 protecting Human Tasks as Critical Resources

While the tasks execution is a critical section for owners, the task itself becomes the critical resource that must be protected accordingly. We can accomplish this very easily by defining the following two operations:

claiming By claiming a task the user becomes the owner of the task.

revoking claim An owner may revoke ownership of a task.

This way we easily can ensure that a task only may have one owner to work on while users with administrative roles may alter the task as well.

Figure 2.14 shows an activity diagram for the two transactions.

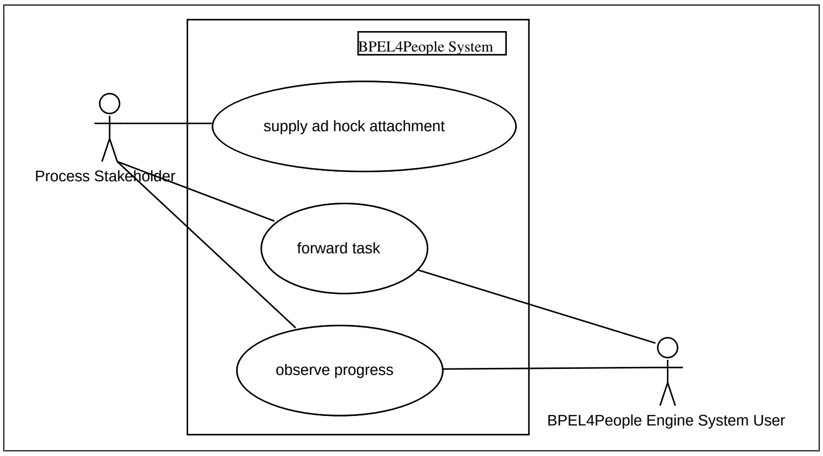

Figure 2.9: Use Case Diagram for a Process Stakeholder

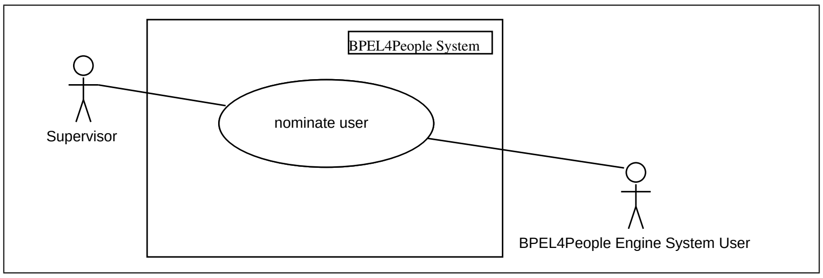

Figure 2.10: Use Case Diagram for a Supervisor

2.4 modelling Scenario

State charts for different scenarios are found in the following tabular use cases corresponding figures.

use case	figure
for eyes principle	2.15
nomination	2.16
escalation	2.17
chained execution	2.18[1]

2.5 defining a Syntax

A BPEL4People syntax has been defined using XML schemata.

Table 2.1 lists and references the corresponding XML schemata.

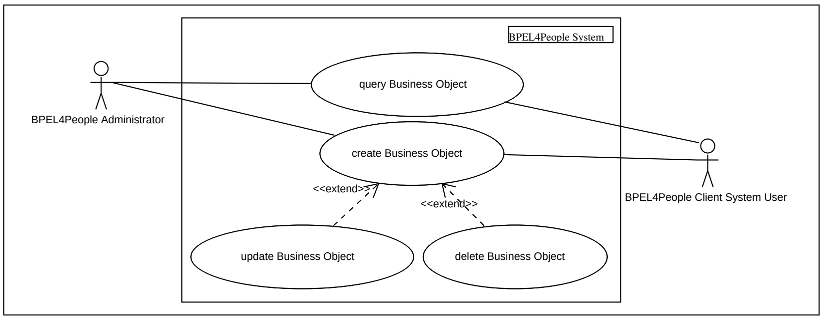

Figure 2.11: generic Use Case Diagram for a BPEL4People Administrator

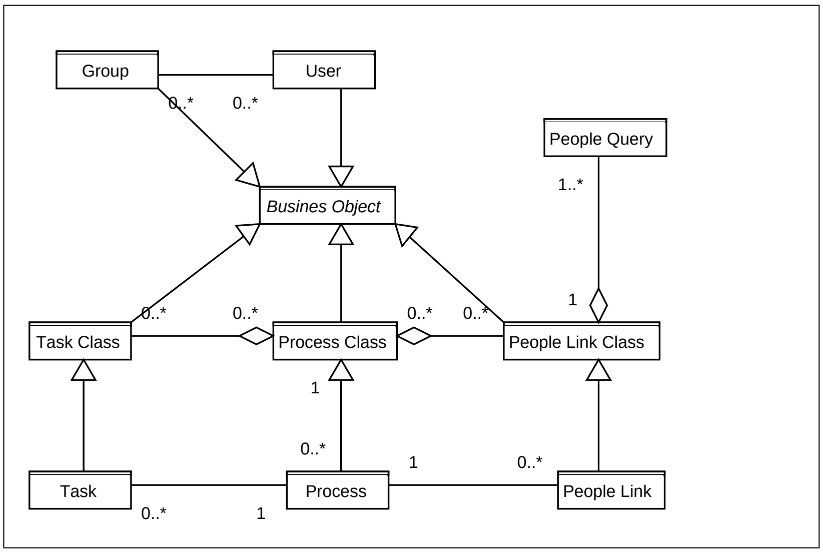

Figure 2.12: Class Diagram for a Business Object

2.6 designing a System

Having specified a proposal for BPEL4People we now want to present our implementation of a BPEL4People system.

As BPEL4People really is an extension to BPEL, we chose to design a system that generically can interact with BPEL engines while hosting BPEL4People specific information and managing BPEL processes on a *People*'s meta level.

Such a BPEL4People system thus will encapsulate a traditional BPEL compliant engine transparently while offering specific interfaces to clients.

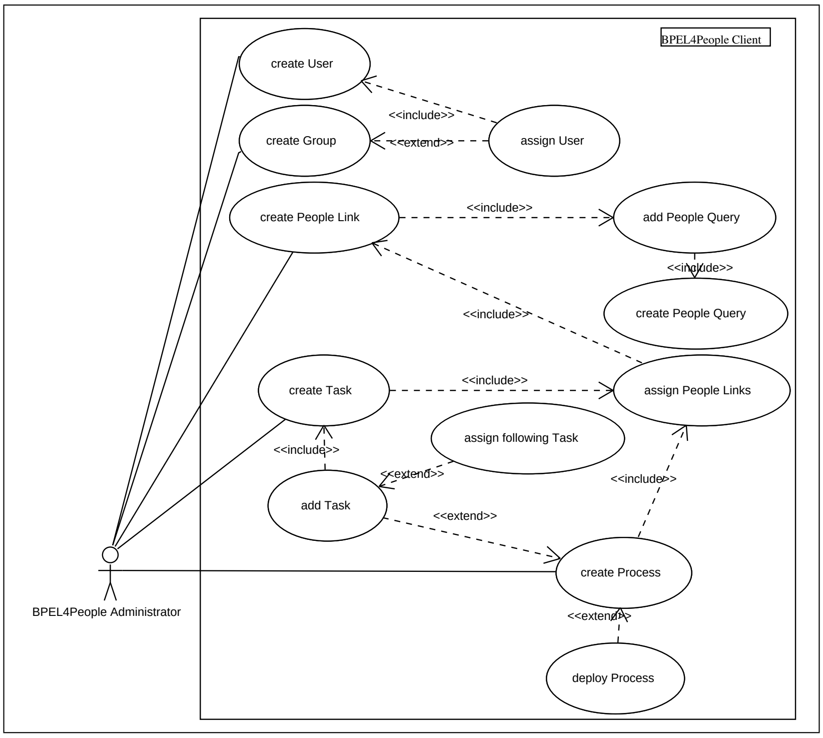

Figure 2.13: Use Case Diagram for a BPEL4People Administrator

We can accomplish that by replacing people activities into a set of invoke and receive activities on the BPEL engine as we will see in section 3.3.9. The people activity is then transparently called by the BPEL engine.

2.6.1 Network Topology

Before describing our architecture of a BPEL4People system, a simple network topology for a system hosting BPEL4People components is shown in figure 2.20.

Web Server

The web server simply forwards requests to an application server and acts thus as a proxy.

Application Server

The BPEL engine is deployed on an application server. Also the BPEL gateway, the engine, the HR service , the notification manager and the web

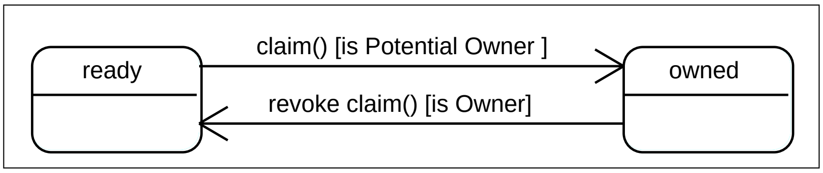

Figure 2.14: State Chart for the Tasks Lock Mechanism

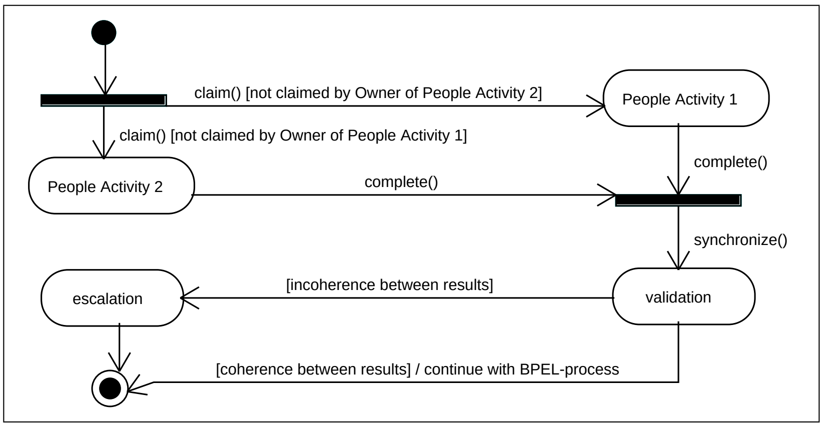
claim() [not claimed by Owner of People Activity 2]
People Activity 1
claim() [not claimed by Owner of People Activity 1]
complete()
People Activity 2
complete()
synchronize()
[incoherence between results]
escalation
validation
[coherence between results] / continue with BPEL-process

Figure 2.15: Activity Graph for the Four Eyes Principle Scenario

client are deployed on application servers that may be distributed over several machines. The BPEL4People web client may receive his requests from a classic web server.

Database Server

The database server stores persistent data for the BPEL as well as for the BPEL4People engine and communicates with the appropriate application servers.

SMS-Gateway

A SMS gateway can be used for sending notifications to BPEL4People users by the notification manager.

HR-Server

A server hosting human resource related information, that can be run individually by an organisation while offering a web service towards the BPEL4People

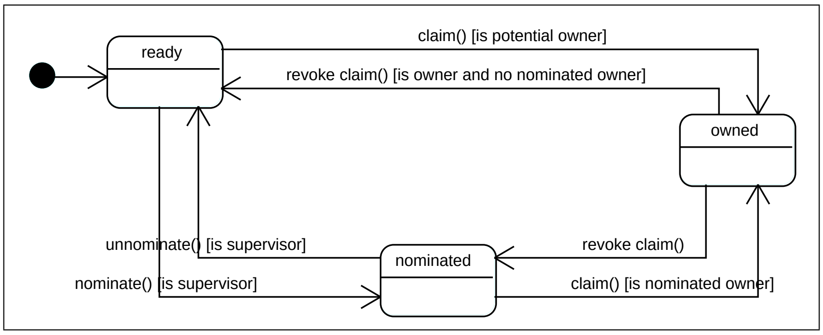
ready
claim() [is potential owner]
revoke claim() [is owner and no nominated owner]
owned
unnominate() [is supervisor]
nominate() [is supervisor]
nominated
revoke claim()
claim() [is nominated owner]

Figure 2.16: State Chart for the Nomination Scenario

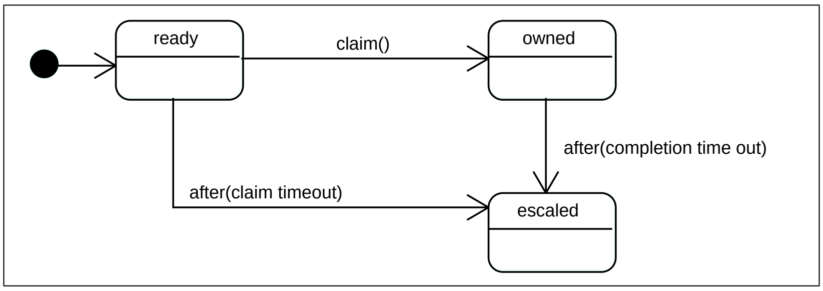

Figure 2.17: State Chart for the Escalation Scenario

Engine for authentication and resolving purposes. We will introduce the HR-server component in 2.6.2.

2.6.2 System Description

- Figure 2.21 shows components of the BPEL4People system.
- Figure 2.22 shows a package diagram for the BPEL4People system.
- Figure 2.23 gives an overview of the systems components by visualising the complete use case's communication.

Web Client

Table 2.2 gives an overview of technologies used by the web client that acts as a user interface for the BPEL4People system to the end user.

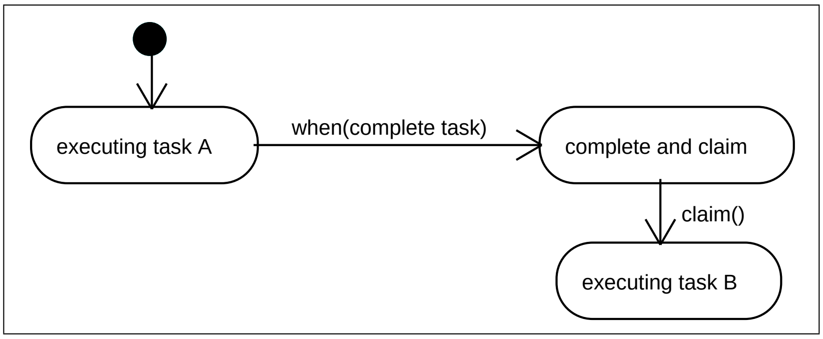

Figure 2.18: Activity Graph for the Chained Execution Scenario

- `http://xml.taid.holmes.at/ns/bpel4people/account.xsd`
- `http://xml.taid.holmes.at/ns/bpel4people/businessobject.xsd`
- `http://xml.taid.holmes.at/ns/bpel4people/process.xsd`
- `http://xml.taid.holmes.at/ns/bpel4people/protocol.xsd`
- `http://xml.taid.holmes.at/ns/bpel4people/task.xsd`

Table 2.1: BPEL4People Syntax and Semantic within XML schemata

Client

Figure 2.24 shows a use case diagram for the client. It implements all clients functions by calling the respective web service operations of the proxy and obviously is used by the web client, that's implementation focuses on it's servlet nature.

Proxy

The proxy is used by the client and is responsible for invoking a web service operation. It does so by simply forwarding the request of the client to the server. The server's response will be returned to the client.

The web service operations for BPEL4People clients are listed in table 2.3.

Server

The server implements the web service for the client. While receiving web service calls the server acts as an application firewall. It does so by intercepting the web service messages and checking for authentication. Also it is

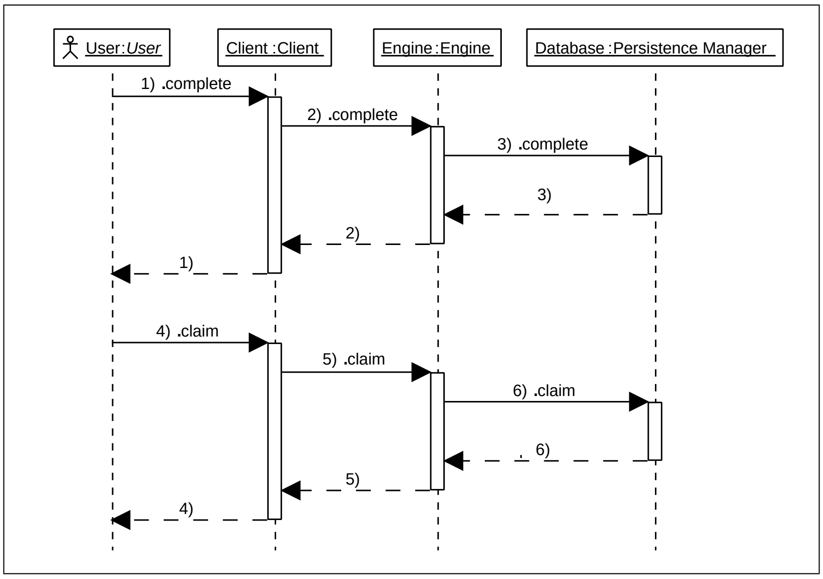

Figure 2.19: Sequence Diagram of a Not-Chained Execution

responsible for logging unauthorised access as well as internal server exceptions in which cases minimalistic information must be given back to clients for security considerations. Authorised calls get forwarded to the engine.

Engine Manager

Regarding the chain of responsibility of the web service operations the engine manager becomes a routing component while the proxy or server simply forwards or blocks requests within the chain.

- login calls are directed towards a HR service
- direct manipulation on business objects are forwarded to the persistence manager.
- indirect manipulation on business objects are handled by the business object tool that we will introduce in section 2.6.2.

BPEL Gateway

A BPEL gateway is responsible for communicating with the BPEL server. It receives invocations for people activities from the BPEL server and activates the corresponding task. If a task completes, the gateway invokes a receive activity on the BPEL engine.

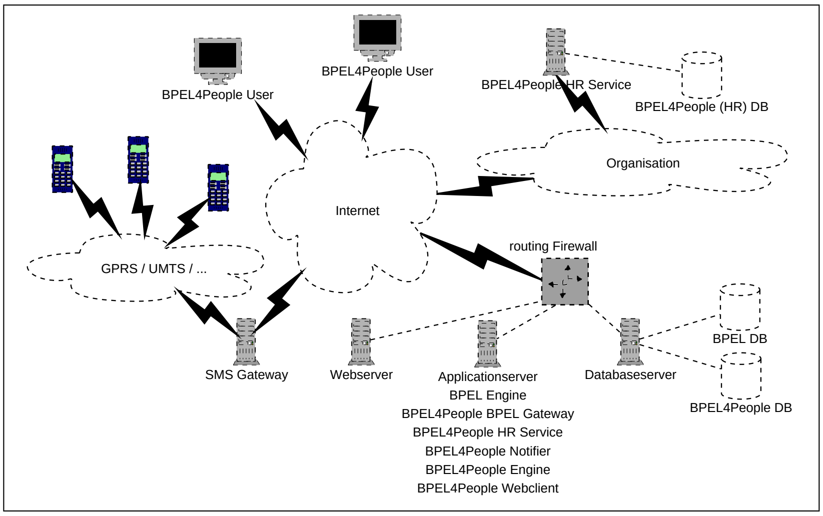

Figure 2.20: Topology of a BPEL4People System

- Hyper Text Markup Language (HTML) 5.1.5
- Cascading Style Sheets (CSS) 5.1.6
- JavaScript (JS) 5.1.7
- Java Server Pages (JSP) 5.1.9
- Spring Framework 5.1.10
- Java Servlets 5.1.11
- Java API for XML-Based Web Services 5.1.12

Table 2.2: Technologies used by the Web Client

HR Server

The HR server returns user objects

- for successful logins
- for people links after resolution
- for a user name

The web service operations for the human resource service are listed in table 2.4.

Query Customiser The query customiser permits customisation of people queries. Individual implementations of customises can be written.

- login
- createBusinessObject
- queryBusinessObject
- updateBusinessObject
- deleteBusinessObject
- queryProcesses
- queryTasks
- deployProcess
- initProcess
- claim
- revokeClaim
- checkin
- complete
- assignPeople
- takeCorrectiveAction
- supplyAttachment
- nominate

Table 2.3: Client Web Service Operations

Deadline Customiser The deadline customiser permits customisation of deadlines. It takes deadline parameters as it's input that can be interpreted in an individually implemented way and returns customised deadlines.

Here are some examples for deadline parameters using a simple grammar, that a customiser would have to interpret:

key	value
owned	1d
completed	owned + 1w
completed	2010-01-02

Business Object Tool

We want to introduce the business object tool that is used for retrieving and manipulating business objects as a peer object.

- login
- isMember
- resolvePeopleLink
- resolveUsername

Table 2.4: HR Web Service Operations

peer object By a peer object we mean an object, that is capable of calling related web service operations autonomously without being necessarily client nor server centred. The mediator software pattern is applied by setting a web service port type as mediator for calling the standard database operations while passing the object.

This tool takes care of revision control by assigning a modified time stamp to the object when it gets committed. Also information as creator or modifier are inherently set for business objects by this generic tool.

Persistence Manager

The persistence manager is responsible of retrieving and manipulating business objects on a persistence layer. The concrete persistence manager is invoked by a factory.

We chose to use a native XML database for storing data using XPath.

Notification Manager

A notification manager can be started together with or independently from the server. Its responsibility is to watch deadlines of tasks and raise escalations in case of a missed deadline. Besides sending notifications to observers the notification manager also creates escalation tasks, sends out escalation notification to escalation recipients of the original task and manipulates the escalation tasks state in case the original task is completed. Escalation tasks are tasks that do not have corresponding people activities. They are fully managed within the manager.

2.6.3 Interfaces

Web Services for Clients and Servers

The following web service descriptions have been elaborated:

Client The web service description of a BPEL4People server for BPEL4People clients.

HR The web service description for the HR service.

Table 2.5 lists and references the corresponding web service descriptions.

- `http://xml.taid.holmes.at/ns/bpel4people/client.wsdl`
  - `http://xml.taid.holmes.at/ns/bpel4people/client.xsd`
- `http://xml.taid.holmes.at/ns/bpel4people/hr.wsdl`
  - `http://xml.taid.holmes.at/ns/bpel4people/hr.xsd`

Table 2.5: BPEL4People web services descriptions

Java Interfaces for Individual Providers

Java interfaces for factories that may be implemented individually include:

Persistence Manager An implementation has been realised for Xindice [40], a native XML database.

BPEL Gateway An abstract class with BPEL engine independent code can be inherited by an actual implementation for a BPEL engine.

Query Customiser A simple XPath customiser was implemented.

Deadline Customiser A default customiser implements the interface in a minimalistic way.

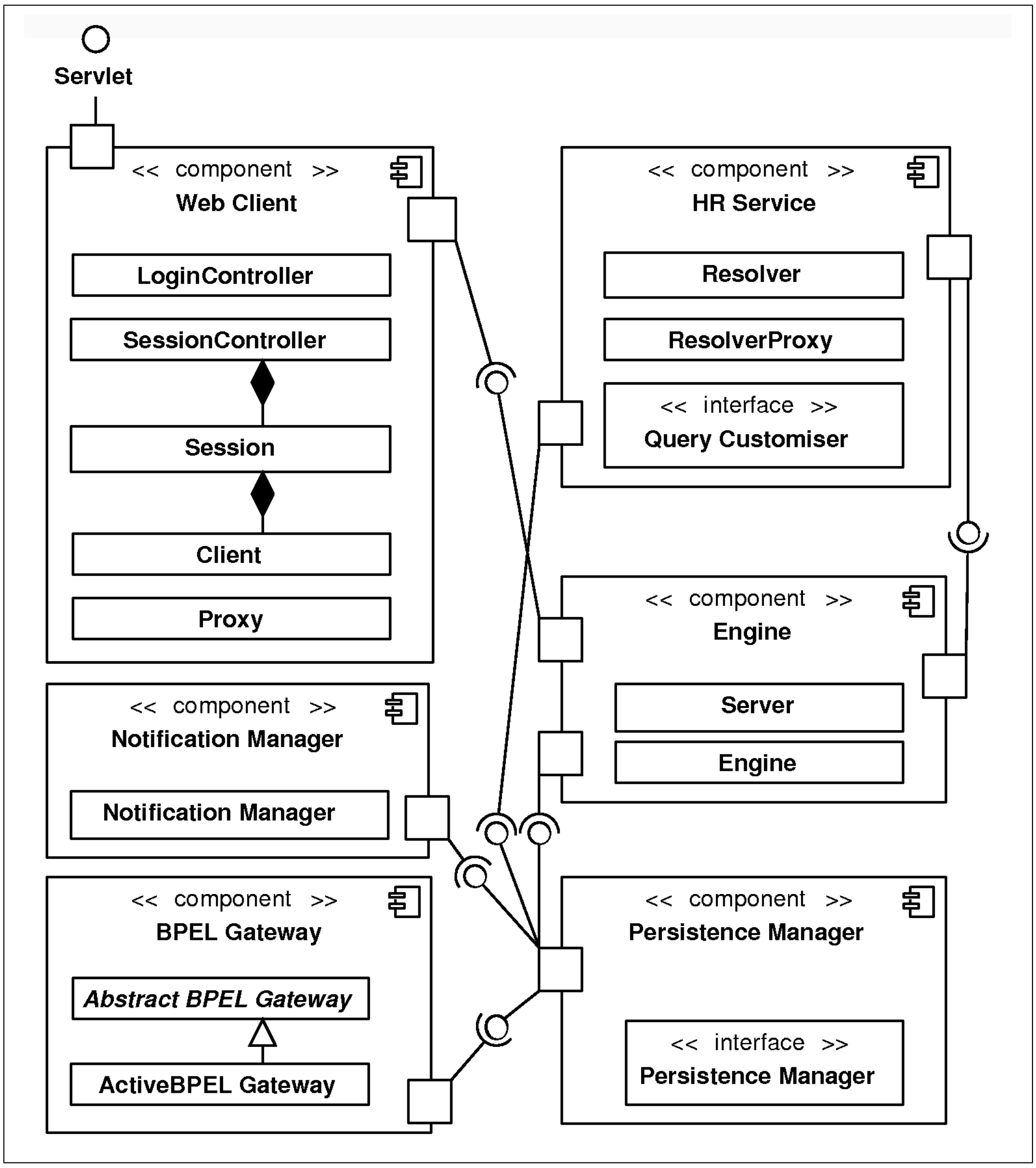

Figure 2.21: Component Diagram for the BPEL4People System

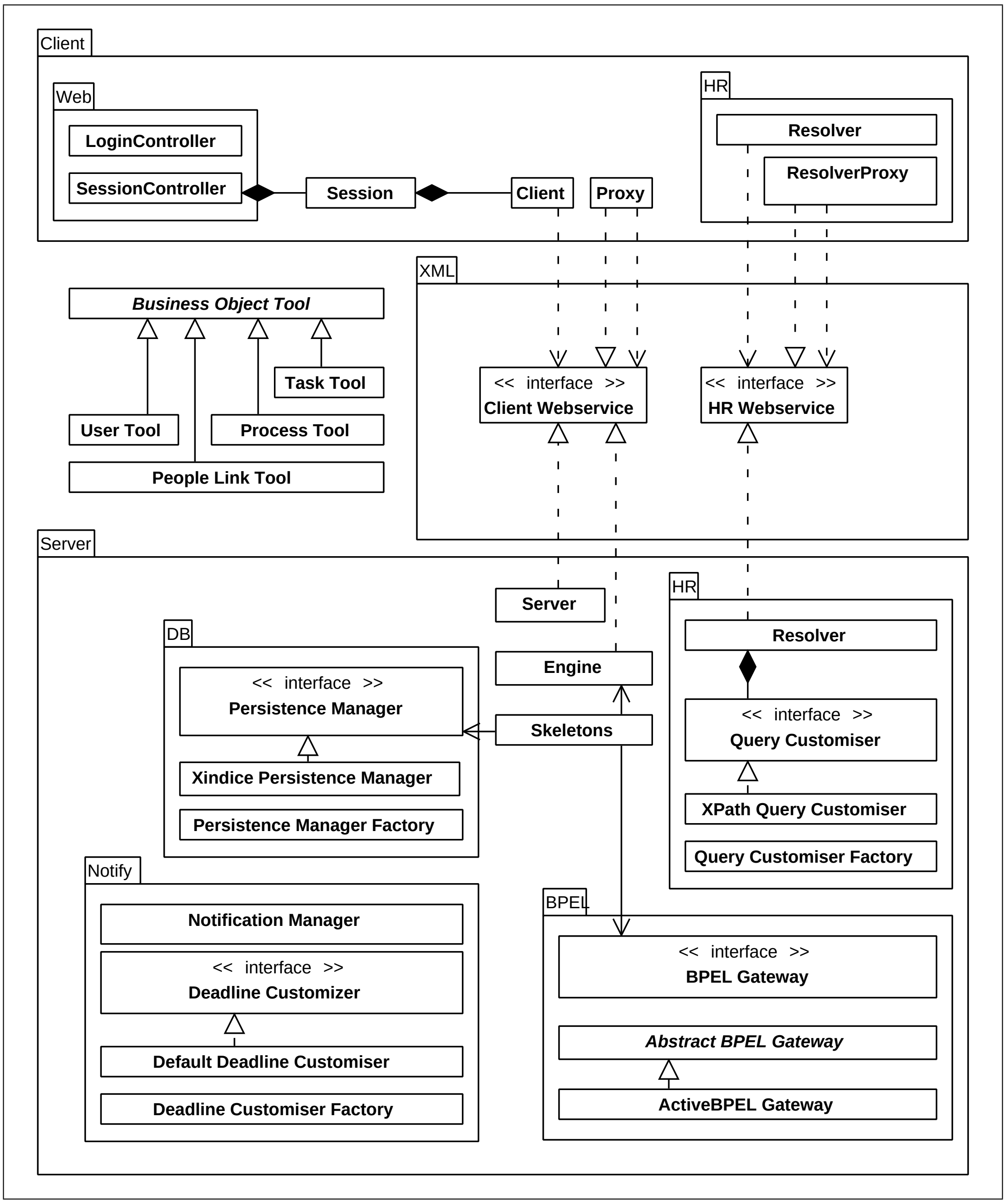

Figure 2.22: Package Diagram for the BPEL4People System

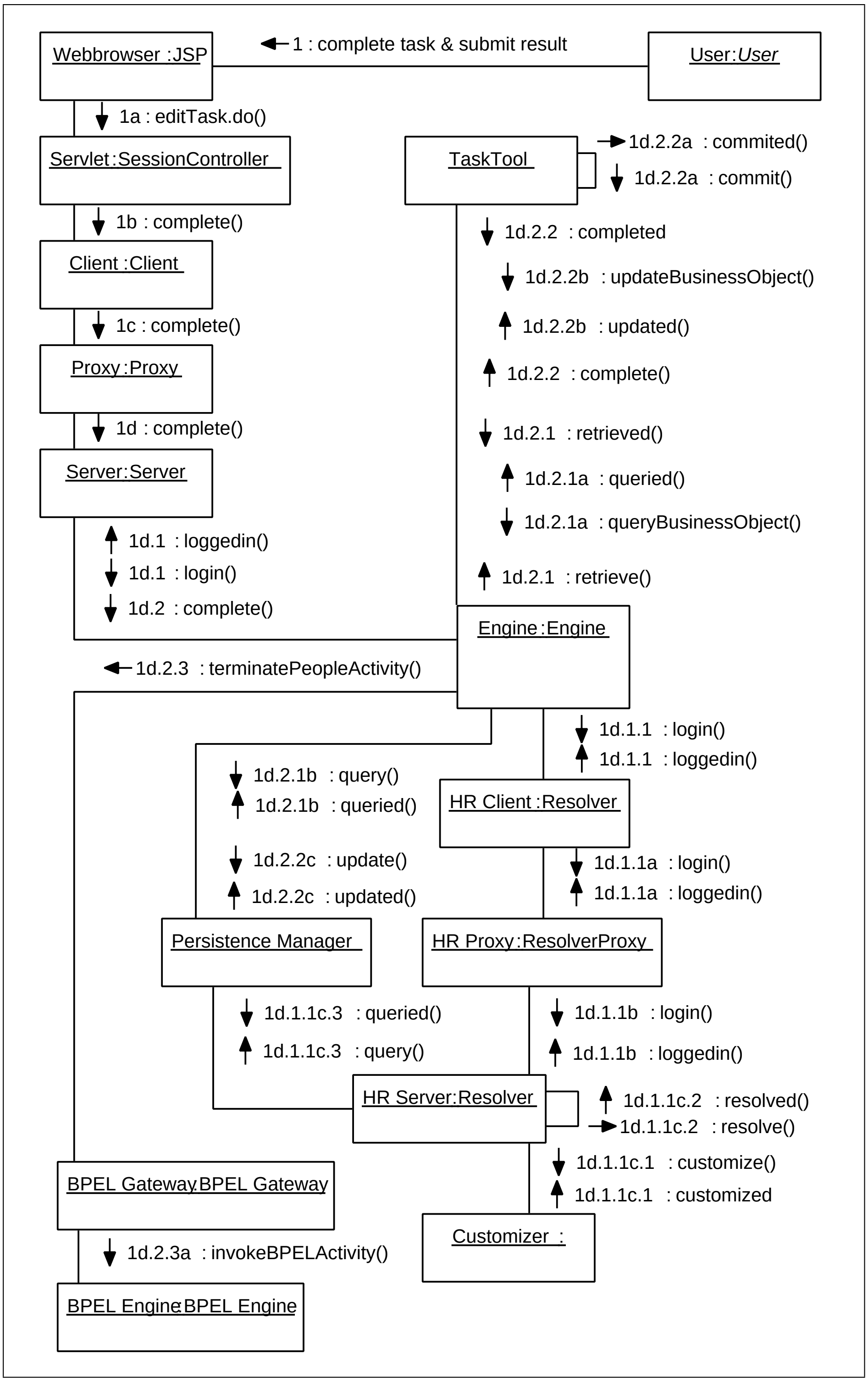

Figure 2.23: Communication Diagram for the Complete Use Case

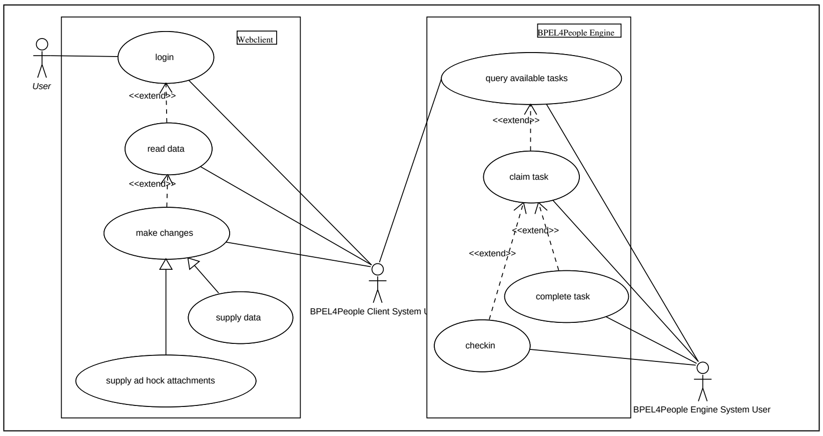

Figure 2.24: Use Cases of the Client

Chapter 3

Discussion

3.1 Validation of Goals

3.1.1 realising Human Process Interaction

Human → Process

instantiation Our system permits (potential) process initiators to initiate new processes that have been defined by BPEL4People administrators. They can do so by assigning people to the role of a process initiator. Furthermore sets of the following parameters can be stated during initiating that will be applied to the process instance for customisation:

resolution parameters Resolution parameters can help customising the people queries that will be interpreted by the HR service. As the implementation of the interpretation of the parameters can be realised in any way there is maximum flexibility regarding the use of these parameters.

deadline parameters Deadline parameters may specify for example during when a task must be claimed or completed. If deadlines are missed escalation takes place. Deadline parameters have to be interpreted by the deadline customiser.

supply data Results of completed tasks or intermediate results can be submitted by the owner of the task. Also it is possible to supply ad hoc attachments.

Human ← Process

request data Data is requested for a task by a process activity. While the input elements can supply an owner with necessary information in order to complete the task, results are given back from the user using output elements.

notification Notification tasks are simple tasks that do not hold output elements.

3.1.2 working exclusively on a Human Task

The human task, as a critical resource, is protected simply by the people activities state so that there must be only one owner if at all. This protection is necessary because we do not want two persons to perform the same task unnecessarily.

3.1.3 assigning People to Roles

While people queries allow the selection of users, people links, containing a set of people queries that are combined set-theoretically, are used to establish the relation between roles and users. While these people links can be reused within processes, people links may contain semantics on the basis of their referencing that indicates their user's role.

3.1.4 supporting Scenario

Nomination

Syntax Nomination is realised by a new role, the nominated owner, that is similar to the one of a potential owner.

System While nominated owners also have to be potential owners in order to be able to successfully claim a task, only nominated owners may do so in case of a nomination.

Four Eyes Principle

Syntax Provision has been made in order to support the four eyes principle scenario that can be realised as following:

The people link's optional attribute useCredits, if greater zero, indicates how often this link may be used[1]. Otherwise there is no limit on how often a role may be executed.

By setting the useRoleCredits attribute of two people activities potential owner's referenced people link to one, the four eyes scenario is realised.

System When a user exercises a role, the people query within the people link is extracted by the BPEL4People system. Its useRoleCredit attribute is then decremented by one and if it becomes zero the attribute operation is set to exclude.

Escalation

Syntax Escalation recipients can be specified in the people links of a task. Moreover the following notification recipients can be specified as people links:

changeObserver A people link that resolved users will be notified of any changes.

statechangeObserver A people link that resolved users will be notified of a state change.

initObserver A people link that resolved users will be notified when a process is initialised.

completionObserver A people link that resolved users will be notified by a tasks completion.

System Conceived as as a additional component for our BPEL4People system a notification manager[2] is polling tasks and checking for missed deadlines. In case of an escalation, a notification manager creates a new escalation task and sends out notifications to escalation recipients. When the original task (that may be an escalation task as well) is completed, the escalation task changes it's status to complete as well. This way unnecessary work that might be performed by escalation recipients can be minimised.

[1] A people link is used when a role's use case is exercised.

[2] 3.3.10

Chained Execution

Syntax Chained execution makes use of a logical link that has to be specified when modelling the process. Concretely people activities that encapsulate tasks can specify a following people activity, using the following attribute for indicating that their tasks are related and that it makes sense to carry them out together.

System This way a BPEL4People client may already complete and claim the following task manually. See figure 2.19 for a pseudo chained execution. With the explicit claimFollowing attribute of the complete element that is sent in a web service message to the server however, the operations of complete and claim may be executed in a transactional way by the server as shown in figure 3.2, ensuring the following task not to get claimed by another potential owner in case this additional security is beneficial.

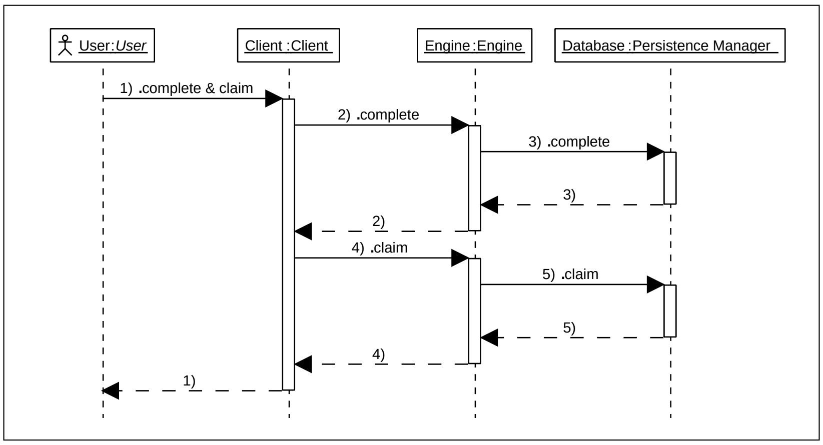

Figure 3.1: Sequence Diagram for a Pseudo Chained Execution

3.2 BPEL4People Syntax

BPEL4People extends BPEL by additional vocabulary that uses it's own namespace. BPEL4People thus can be mapped to ordinary BPEL by extracting all BPEL4People specific data and transforming people activities into ordinary BPEL activities. The resulting BPEL process can be deployed on a BPEL engine while a BPEL4People system coordinates the communication with it while interpreting BPEL4People specific data.

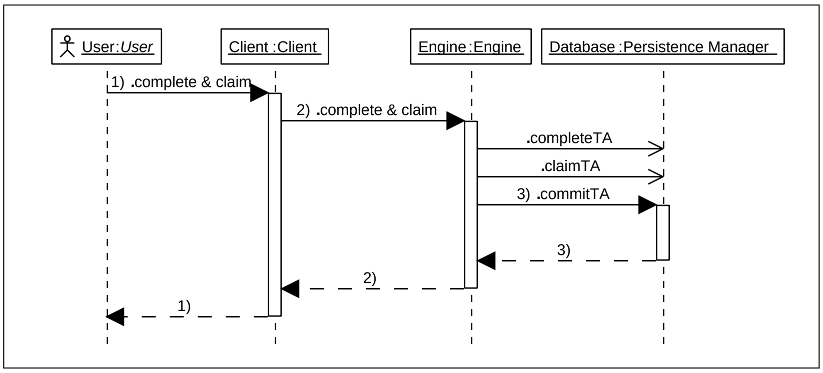

Figure 3.2: Sequence Diagram for the Chained Execution Scenario

3.3 BPEL4People System

3.3.1 Web Client

Client

The web client consisting of just two JSP pages uses CSS and java script technologies on the client side. Using these technologies the web client becomes interactive and partially client-centred. A java script stack of HTML div "windows" is implemented for the web client that windows can overlapped, toggled to full screen, etc. Also dialogues wizards are realised using these technologies. Verification of input takes place when submitting data. A password for example, that has to be entered twice is type error checked already by the client. HTTP requests are minimised this way.

After successful login, the web client customises the menu that will be represented to users. While only BPEL4People administrators can manage users, groups, and classes of people links, tasks as well as processes only users possessing a process initiator role for a selected process will be given the option to initiate it.

Server

The web client is a server – client application. A multi-action session controller receives and processes data from the client. At this level already we use Java Beans that are enriched by JAXB in order to directly map them to XML.

3.3.2 Client

The separation of web client and client is that servlet dependant code remains within the web client while servlet independent code that can be relevant to any - not necessarily a web client is placed at the client. The client formulates the request using passed data.

3.3.3 Server

The server implements responsibility for

access control The server verifies if a caller possesses the necessary role in order to execute web services operation.

masquerade exceptions Exceptions that occur within the BPEL4People engine are caught and masqueraded for security reasons in order not to disclose internal information of the engine.

logging The server may log information for authentication requests as well as for documenting internal exceptions.

3.3.4 Engine

The engine

routes operations Database operations are forwarded to the persistence manager while use cases are prepared to be executed

triggers use case execution Use cases are executed using the appropriate business object tool.

calls the BPEL gateway in case a tasks completes.

3.3.5 HR Service

Such a server can implement individual customisation that can be applied during people link resolution. This brings the following advantages:

- Sensitive and individual related data can be directly hosted by an organisation and does not have to be placed externally.
- A huge - for instance international - organisation that might run it's own BPEL4People engine might choose to delegate the BPEL4People HR service towards it's national branches that run their respective BPEL4People HR server autonomously.

- Information as the locality of the HR server or other criteria can be applied for people resolution permitting individual customisation.

3.3.6 Persistence Manager

The persistence managers implementation supports standard database operations using a concrete database for storing business objects.

3.3.7 Business Object Tools

A business object tool composites a business object, that holds BPEL4People relevant data.

revision control Business object tools implicitly and inherently apply revision control to its business objects.

use case execution Concrete business object tools implement specific use cases.

account Does simple accounting using the useCredits attribute of people links

3.3.8 HR Service

The HR service hides the complexity of people links and queries toward callers. Everything higher than user objects is managed by this service. It permits customisation of people queries by offering an extension point for interpreting resolution parameters. The HR service can fully be delegated to a third party and does not have to reside on the engine.

3.3.9 BPEL Gateway

Concrete BPEL gateways for BPEL engines have to be provided that

- deploy BPEL to a specific BPEL engine. Using the factory pattern a specific implementation can be realised for a certain BPEL engine in a plugin fashion.

- receives BPEL invocation for people activities and sets its tasks input and its state to ready.

- invokes a BPEL reply, when an activity completes passing outputs.

Deployment

The BPEL gateway receives BPEL code to deploy on a BPEL server where BPEL4People specific information but the the people activity has been extracted. People Activities are transformed into a pair of invoke and receive activities.

A process as shown in table 3.1 would finally be transformed into code as shown in table 3.2 for final deployment on a BPEL engine.

```
<process name="ID.NAME">
  ...
  <partnerLinks>
    ...
  </partnerLinks>
  ..
  <people>...</people>
  ...
</process>
```

Table 3.1: BPEL4People extracted BPEL with people activities

```
<process name="BPEL4People.ID.NAME">
  ...
  <partnerLinks>
    <partnerLink name="bpel4peopleGENERATED"
      partnerLinkType="bpel4PeopleLTGENERATED"
      partnerRole="bpel4peopleBPELGateway"
      myRole="bpel4peopleBPELEngine"/>
  </partnerLinks>
  ..
  <invoke partnerLink="bpel4peopleGENERATED"
    operation="invokePeople"
    inputVariable="invokePeopleMessage">
    ...
  </invoke>
  <receive partnerLink="bpel4peopleGENERATED"
    operation="invokedPeople" inputVariable="invokedPeopleMessage">
    ...
  </receive>
  ...
</process>
```

Table 3.2: BPEL4People transformed BPEL

In order to do so a corresponding WSDL[3]has to be generated, that defines

partnerLinkType The partner link type to set up as a channel between the BPEL gateway and the BPEL engine for the people activity.

invokePeopleMessage The message, encapsulating BPEL4People inputs, to be passed as input variable to the BPEL gateway.[4]

invokedPeopleMessage The message, encapsulating BPEL4People outputs, to be received as input variable from the BPEL gateway.[4]

Invocation of a People Activity

Invocation of a people activity takes place by

- passing BPEL4People inputs as well as a
- reference to the task id within the message and
- calling the BPEL gateways web service

The BPEL4People inputs are taken from the message for the the referenced task.

Termination of a People Activity

When terminating a people activity the BPEL gateway

- maps the task's id to its corresponding partner link.
- passes the BPEL4People outputs to the message.
- calls the BPEL engines web service.

3.3.10 Notification Manager

Our BPEL4People engine can be extended by a server's independent notification manager that can make use of our persistence manager for accessing the BPEL4People database to operate on. The notification manager is a fully independent component from the rest of the BPEL4People system that may be deployed anywhere having access to the BPEL4People database nonetheless.

[3] template: `http://xml.taid.holmes.at/ns/bpel4people/gateway.wsdl`.

[4] `http://xml.taid.holmes.at/ns/bpel4people/gateway.xsd`

3.3.11 Software Architecture

Within our BPEL4People system the following architectural software patterns [13] exist:

Creational Patterns

Singleton We use static instances of objects for the engine, the BPEL gateway and the persistence manager.

Factory This software pattern is applied for the persistence manager as well as the query and deadline customiser. This way independent implementations of for example various BPEL engines or databases can transparently be realised.

Also web services descriptions can be regarded as interfaces of factories that moreover permit interoperability and distribution.

Structural Patterns

Composite Our XML data structure contains composite elements. Also corresponding java beans generally do composite structures.

Proxy The proxy is used by the web client's client.

Behavioural Patterns

Chain of Responsibility Focus is given to this behavioural pattern as it permits requests to be handled at various levels. This might be due to security reasons or because of performance considerations. A client – server request thus not only consists of two but various software components with each component having a specific responsibility how to handle the request.

Mediator A variation of the mediator pattern is applied at business object tools. A port type that is implemented by various components within the chain of responsibility usually is set acting as the mediator towards the tool.

Plugin Customisers are implemented as plugins using the factory pattern. A plugin[5] can thus be done by implementing the interface.

[5] term used within the factory pattern: product

Chapter 4

Summary

4.1 Syntax

While BPEL can be used to describe business processes, we have enriched BPEL by additional orthogonal syntax and semantics resulting in BPEL4People so that:

- human tasks are protected as critical resources from potential owners
- assignment of people to roles have been conceived
- BPEL4People scenario can be supported by a BPEL4People system interpreting BPEL4People

4.2 System

BPEL4People, as it says in it's name, is there for people (to work with).

Our syntax definition and our system:

- enables users to specify BPEL4People definitions
  - by assigning people to roles
  - by creating tasks for BPEL process activities, that are encapsulated by people activities
  - by creating processes
- enables users to work with BPEL4People instances
  - by instantiating processes

    - by querying tasks and processes
    - by altering a tasks state
    - by submitting work and ad hoc attachments
- interacts proactively with users
    - by notifying users of changes or events[1]

Moreover, on a more technical note, it

- hosts people activities
    - that encapsulate human tasks
    - that are invokable transiently as common activities by BPEL processes and engines[2]
- interacts with a BPEL engine
    - by managing the people activities state and result within the context of the process
    - by deploying a BPEL process to it[2], while extracting and conserving BPEL4People specific information, that will be associated with the process
    - by handling inputs and outputs[2]

4.3 Further Work & Research

4.3.1 Syntax

Access Control

Access control is realised implicitly using generic human roles. The access of the revoke claim operation for example is controlled by the server manually and explicitly so that only the tasks owner may perform it. This logic thus is hard coded at the server. It would be interesting to migrate this logic into the BPEL4People syntax. This way BPEL4People would not be limited to it's defined generic human roles. New type of roles could be defined dynamically as well as corresponding access controls. For this to happen BPEL4People web service operations have to be mapped to people links.

[1] when extended by a notification manager

[2] when extended by a concrete BPEL gateway implementation

4.3.2 System

Web Client

Localisation Tasks can be described multilaterally. So far when querying tasks all supplied languages are transferred to the client. Our web clients servlet may select the browsers preferable language by evaluating the X-Accept-Language HTTP header.

Interactivity Our web application is highly interactive thanks to java script and CSS. It has been optimised for minimal data transfer and new HTTP requests are only made when submitting data. Complex dialogues are managed by dynamic HTML, JSP and the spring framework. Nonetheless AJAX [14] can be integrated into the web client additionally too. For instance it might be interesting to receive preview information using over-mouse events of interesting elements.

Human Roles & User Interfaces Although our web client can be generically used for any role, it is optimised for administration nonetheless. Particularly the user interfaces that may be specified with a task are not interpreted yet. This is because a standard has to be defined how these user interfaces should be used and how the binding of input data takes place. XForms [45] was suggested for these user interfaces by [1] as a technology. The topic of customised user interfaces for BPEL4People needs to be addressed properly.

Engine

Localisation Localisation may take place between a client and the server already. The web client then would have to pass preferable languages to the server that filters tasks descriptions out accordingly.

Load Balancing Load balancing of a BPEL4People system might be an interesting topic to investigate.

Accounting The useLink function within the people process tool can be extended for integrating a plugin that does accounting.

Persistence Manager

Distributed databases may replace a local one. The persistence manager then is not exclusively using the database anymore. Simple as well as complex transactions will then have to be implemented using a databases functionality only.

Notification Manager

A notification manager has to be implemented for our BPEL4People system. While notification can be realised by a polling web client, a push system has to be developed. Particularly notification by SMS or email have to be supported. BPEL4People users must be able to subscribe to processes or tasks.

BPEL Gateway

A BPEL engine specific implementation of a BPEL gateway has to be provided in order to use our BPEL4People system with the respective BPEL engine.

4.3.3 BPEL

While BPEL realises web service orchestration, hiding the complexity of composed web services as an ordinary web service and thus permits expansion of a distributed process model, each part of an orchestration is fixed. While the process model can be redesigned, process instances cannot be altered. However intelligent exception handling can be done within BPEL [16]. Research needs to be done if a large number of process instances work flow have to be altered or if design errors have occurred [30].

4.3.4 General Remarks

Security

Security should be applied on every level possible within a distributed system. Public key cryptography technologies as for example transport layer security (TLS) [20] should be applied between a web clients browser and the web server as well as between the web service client and the server [10] for privacy and integrity of data.

Testing

Besides the testing of BPEL processes [28], it is obvious that there is a immense number of possible combinations regarding given roles, processes, tasks and notifications so that a systematic testing has to be undertaken for a BPEL4People system. As this is not an easy task to do in a distributed system, it is interesting to apply research in the field of distributed testing as [11] to our system.

Chapter 5

Materials & Background

5.1 Theoretical Background

BPEL4People [1], an extension to BPEL, is layered on top of several specifications. Figure 5.1 gives a simple dependency overview of XML sets while figure 5.2 shows BPEL4People in the context of the OSI model [21].

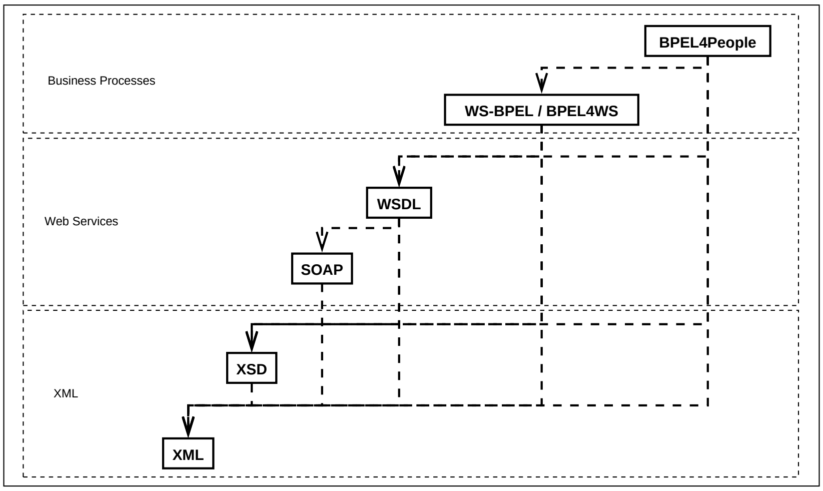

Figure 5.1: Dependency Graph of XML & BPEL4People

5.1.1 Extensible Markup Language

The extensible markup language (XML) [5], a simplified version of the standard generalised markup language (SGML) [25], organises information in a tree based structure. A XML document has to be well-formed in order to

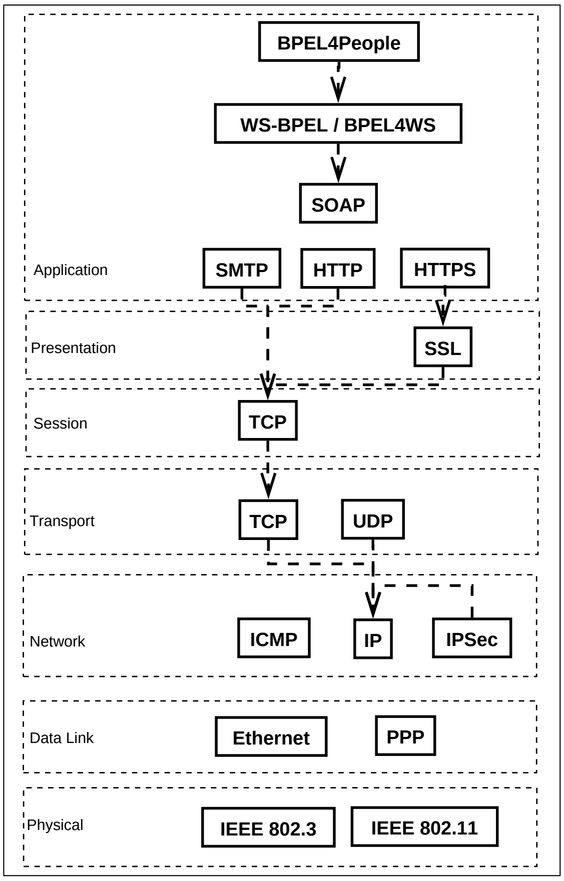

Figure 5.2: OSI Model & BPEL4People

be processed by a XML parser. It can moreover be valid in which case the document conforms to a corresponding XML schema.

BPEL4People is a subset of XML.

5.1.2 XML Schema

The XML schema standard defines structures [44] as well as data types [4] for describing a class of valid XML documents. XML schema creates XML vocabularies that meaning can be documented and that use can be constrained within documents.

BPEL4People is defined with XML schema.

- XML declaration
- element
- attribute

Table 5.1: XML Components

- attribute
- element
- simpleType
- complexType
- attributeGroup
- annotation

Table 5.2: XML Schema Components

5.1.3 Web Services

[31] gives the following definition of a web service:

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialisation in conjunction with other Web-related standards.

Major BPEL4People system components interfaces have been realised as web services. WS-BPEL furthermore is designed for Web services.

Web Services Description Language

Web Services Description Language (WSDL) [41], a XML subset, describes web services by defining it's interfaces.

The WSDL 1.1 specification defines components as listed in 5.3.

All important BPEL4People system's interfaces are defined by web services descriptions.

types a container for data type definitions using some type system (such as XSD).

message an abstract, typed definition of the data being communicated.

operation an abstract description of an action supported by the service.

portType an abstract set of operations supported by one or more endpoints.

binding a concrete protocol and data format specification for a particular port type.

port a single endpoint defined as a combination of a binding and a network address.

service a collection of related endpoints.

Table 5.3: WSDL Components

5.1.4 WS-BPEL

Business Process Execution Language

The business process execution language (BPEL) is a XML subset defining business processes [32] [3]. As interactions are realised with web services for maximum interoperability between various heterogeneous systems, BPEL permits orchestration of web services.

Processes consist of activities like the following that correspond to work flow tasks:

Correlation Sets

Correlation Sets are used for routing messages to a specific process instance within BPEL.

Basic Activities Table 5.4 shows basic BPEL activities.

The people activity that is realised by BPEL4People is transparently represented as an ordinary BPEL activity towards the BPEL server that is normally invoked using an invoke activity.

Structured Activities Table 5.5 shows structured BPEL activities.

Scopes

An activity may be wrapped within a scope that can define data variables, correlation sets, fault handlers, a compensational handler, as well as event handlers. A scope can be set to be serialisable. In such a case it must not include nested scopes.

- invoke
- receive
- reply
- assign
- throw
- terminate
- wait
- empty
- sequence
- switch
- while
- pick
- flow
- scope
- compensate

Table 5.4: BPEL Basic Activities

Compensation

Activities may define compensation handlers that include a compensational activity.

[8] describes exception handling in BPEL4WS.

Partner Links

BPEL integrates WSDL port types by associating them with roles. Two roles with their implicit port types finally give a partner link that characterises the relationship between the two services.

When deploying a BPEL4People process to a BPEL engine containing a people activity the partner links are used to setup a communication channel with the engine.

Extensibility

BPEL4People extends BPEL by introducing constructs that make use of dedicated a XML namespace. A BPEL4People engine must thus support the BPEL4Peoplés namespace.

- empty
- sequence
- switch
- while
- pick
- flow

Table 5.5: BPEL Structured Activities

WS-BPEL 2.0 Section 14 in [32] declares extensibility for WS-BPEL by using a foreign namespace. An extension element can be used to indicate a BPEL engine that it must support a foreign XML namespace's WS-BPEL extension.

```
<extension namespace="anyURI" mustUnderstand="yes|no" />+
```

With this construct a BPEL engine could easily be made BPEL4People aware. This is to say it may invoke a deployment on the BPEL4People engine, when it recognises the appropriate extension.

BPEL4WS Section 6.3 of [3] explicitly permits the extensibility of the language by simply using constructs from other XML namespaces.

5.1.5 Hyper Text Markup Language

The hyper text markup language (HTML) [43] is an SGML [25] application, and is widely regarded as the standard publishing language of the World Wide Web.

Our web client is XHTML1.0 valid.

5.1.6 Cascading Style Sheets

Cascading style sheets (CSS) [42] is a style sheet language that allows authors and users to attach style to structured documents. By separating the presentation style of documents from the content of documents, CSS simplifies web authoring and site maintenance.

Our web client is CSS2 valid.

5.1.7 JavaScript

Java script (JS) [37] is an object-oriented programming language for performing computations and manipulating computational objects within a host environment.

Java script, together with XHTML and CSS, realises a simple window system at the web client. Java script is also used to validate user input on the web client.

5.1.8 Java Beans

A java bean [38] is a reusable software component that can be manipulated visually in a builder tool.

Java beans are used with our web clients java server pages. All classes that has been generated from XML source are java beans.

5.1.9 Java Server Pages

Java server pages (JSP) [35] is a technology for building applications for generating dynamic web content, such as HTML, DHTML, XHTML, and XML.

Few of the plain JSP functionality is used without the Spring Framework. Also some of the dynamic content is realised by the client itself.

5.1.10 Spring Framework

Spring is a layered Java/J2EE application framework, based on code published in [24].[1]

The JSP sites make use of the spring framework. Java beans containing form input data are handled by these technologies.

5.1.11 Java Servlets

A servlet is a JavaTM technology-based Web component, managed by a container, that generates dynamic content. Servlets interact with web clients via a request/response paradigm implemented by the servlet container [7].

[1] http://www.springframework.org

Our web client contains a multi-action controller that processes input data using the spring framework.

5.1.12 Java API for XML-Based Web Services

The java API for XML-based web services (JAX-WS) defines APIs and conventions for supporting RPC oriented XML web services in the java™ platform [6].

Most notably the java XML bindings JAXB are part of this API. JAXB is used for transforming java beans to XML that can directly be stored in a native XML database. The java beans correspond to XML complex type structures are generated from XML schema itself and used in all BPEL4People's components from the database up to the JSP's form input elements.

5.2 Unified Modelling Language

The unified modelling language (UML) [22] is used in this document for

5.2.1 Structure Diagrams

class diagrams A class diagram is a collection of static declarative model elements, such as classes, interfaces, and their relationships, connected as a graph to each other and to their contents. [22]

5.2.2 Behavior Diagrams

use case diagrams Use case diagrams show actors and use cases together with their relationships. The use cases represent functionality of a system or a classifier, like a subsystem or a class, as manifested to external interactors with the system or the classifier. [22]

state charts A statechart diagram can be used to describe the behaviour of instances of a model element such as an object or an interaction. [22]

activity diagrams An activity diagram is a special case of a state diagram in which all (or at least most) of the states are action or subactivity states and in which all (or at least most) of the transitions are triggered by completion of the actions or subactivities in the source states. [22]

5.2.3 Interaction Diagrams

collaboration diagrams A collaboration diagram presents either a Collaboration, which contains a set of roles to be played by Instances, as well as their required relationships given in a particular context, or it presents a CollaborationInstanceSet with a collection of Instances and their relationships. The diagram may also present an Interaction (InteractionInstanceSet), which defines a set of Messages (Stimuli) specifying the interaction between the Instances playing the roles within a Collaboration to achieve the desired result. [22]

sequence diagrams A sequence diagram presents an Interaction, which is a set of Messages between ClassifierRoles within a Collaboration, or an InteractionInstanceSet, which is a set of Stimuli between Instances within a CollaborationInstanceSet to effect a desired operation or result. [22]

5.2.4 Implementation Diagrams

component diagrams Component diagrams show the structure of components, including the classifiers that specify them and the artifacts that implement them. [22]

Chapter 6

Acknowledgements

- Prof. Schahram Dustdar
- DI Martin Vasko
- Sylvester Keil

Chapter 7

References

Bibliography

[1] a joint IBM-SAP whitepaper. *WS-BPEL Extension for People*, August 2005.

[2] Anzböck, R., Dustdar, S., and Gall, H. Software configuration, distribution, and deployment of web-services. In *SEKE '02: Proceedings of the 14th international conference on Software engineering and knowledge engineering* (New York, NY, USA, 2002), ACM Press, pp. 649–656.

[3] BEA Systems, IBM, Microsoft, SAP AG and Siebel Systems. *Business Process Execution Language for Web Services Version 1.1*, May 2003.

[4] Biron, P. V., and Malhotra, A. *XML Schema Part 2: Datatypes*, October 2004.

[5] Bray, T., Paoli, J., and Sperberg-McQueen, C. M. *Extensible Markup Language (XML) 1.0*, February 1998.

[6] Chinnici, R., Hadley, M., and Mordani, R. *The Java API for XML-Based Web Services (JAX-WS) 2.0*, April 2006.

[7] Coward, D., and Yoshida, Y. *Java™ Servlet Specification Version 2.4*, November 2003.

[8] Curbera, F., Khalaf, R., and Frank Leymann, S. W. Exception handling in the bpel4ws language. In *Business Process Management* (2003), pp. 276–290.

[9] Curbera, F., Khalaf, R., Mukhi, N., Tai, S., and Weerawarana, S. The next step in web services. *Commun. ACM 46*, 10 (2003), 29–34.

[10] Damiani, E., di Vimercati, S. D. C., and Samarati, P. Towards securing xml web services. In *XMLSEC '02: Proceedings of the 2002*

ACM workshop on XML security (New York, NY, USA, 2002), ACM Press, pp. 90–96.

[11] Dustdar, S., and Haslinger, S. Testing of service oriented architectures – a practical approach. In *Object-Oriented and Internet-Based Technologies* (December 2004).

[12] Ferratt, T. W., Lederer, A. L., Hall, S. R., and Krella, J. M. Information technology and competitors: a case for collaborative advantage. In *SIGCPR '95: Proceedings of the 1995 ACM SIGCPR conference on Supporting teams, groups, and learning inside and outside the IS function reinventing IS* (New York, NY, USA, 1995), ACM Press, pp. 139–147.

[13] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. *Design Patterns: Elements of reusable object oriented software.* Addison Wesley Publishing Company, 1994.

[14] Garrett, J. J., February 2005.

[15] Georgakopoulos, D., Hornick, M., and Sheth, A. An overview of workflow management: from process modeling to workflow automation infrastructure. *Distrib. Parallel Databases 3*, 2 (1995), 119–153.

[16] Golani, M., and Gal, A. Optimizing exception handling in workflows using process restructuring. In *Business Process Management* (2006), pp. 407–413.

[17] Guelfi, N., and Mammar, A. A formal framework to generate xpdl specifications from uml activity diagrams. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing* (New York, NY, USA, 2006), ACM Press, pp. 1224–1231.

[18] Hamadi, R., and Benatallah, B. A petri net-based model for web service composition. In *ADC '03: Proceedings of the 14th Australasian database conference* (Darlinghurst, Australia, Australia, 2003), Australian Computer Society, Inc., pp. 191–200.

[19] IBM Software Group. *Web Services Flow Language (WSFL) version 1.0*, 2001.

[20] The Internet Engineering Task Force. *The Transport Layer Security (TLS) Protocol Version 1.1 (RFC 4346)*, April 2006.

[21] ISO/IEC. *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*, November 1994.

[22] ISO/IEC. *Unified Modeling Language (UML), v1.4.2*, April 2005.

[23] Ives, B., and Learmonth, G. P. The information system as a competitive weapon. *Commun. ACM 27*, 12 (1984), 1193–1201.

[24] Johnson, R. *Expert One-on-One J2EE Design and Development.* Wiley & Sons, October 2002.

[25] JTC 1/SC 34. *Information processing – Text and office systems – Standard Generalized Markup Language (SGML)*, March 1986.

[26] Lohmann, N., Massuthe, P., Stahl, C., and Weinberg, D. Analyzing interacting bpel processes. In *Business Process Management* (2006), pp. 17–32.

[27] Martin, V., and Dustdar, S. A view based analysis of workflow modeling languages. In *Lecture Notes in Computer Science* (2006), pp. 276–290.

[28] Mayer, P., and Lübke, D. Towards a bpel unit testing framework. In *TAV-WEB '06: Proceedings of the 2006 workshop on Testing, analysis, and verification of web services and applications* (New York, NY, USA, 2006), ACM Press, pp. 33–42.

[29] Medjahed, B., Benatallah, B., Bouguettaya, A., Ngu, A. H. H., and Elmagarmid, A. K. Business-to-business interactions: issues and enabling technologies. *The VLDB Journal 12*, 1 (2003), 59–85.

[30] Modafferi, S., Mussi, E., and Pernici, B. Sh-bpel: a self-healing plug-in for ws-bpel engines. In *MW4SOC '06: Proceedings of the 1st workshop on Middleware for Service Oriented Computing (MW4SOC 2006)* (New York, NY, USA, 2006), ACM Press, pp. 48–53.

[31] Note, W. W. G. *Web Services Architecture (WSA).*

[32] OASIS Web Services Business Process Execution Language (WSBPEL) TC. *Web Service Business Process Execution Language Version 2.0*, January 2007.

[33] Open Management Group. *MDA Guide Version 1.0.1*, June 2003.

[34] Open Management Group. *Business Process Modeling Notation Specification*, May 2006.

[35] Roth, M., and Pelegrí-Llopart, E. *JavaServer Pages™ Specification Version2.0*, November 2003.

[36] Russell, N., van der Aalst, W. M. P., ter Hofstede, A. H. M., and Wohed, P. On the suitability of uml 2.0 activity diagrams for business process modelling. In *APCCM '06: Proceedings of the 3rd Asia-Pacific conference on Conceptual modelling* (Darlinghurst, Australia, Australia, 2006), Australian Computer Society, Inc., pp. 95–104.

[37] Standard ECMA-262. *ECMAScript Language Specification*, December 1999.

[38] Javabeans™, August 1997.

[39] Thatte, S. *Web Services for Business Process Design.* Microsoft Corporation, 2001.

[40] Xindice.

[41] W3C Note. *Web Services Description Language (WSDL) Version 1.1.*

[42] W3C Recommendation. *Cascading Style Sheets, level 2 CSS2 Specification*, May 1998.

[43] W3C Recommendation. *XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition)*, August 2002.

[44] W3C Recommendation. *XML Schema Part 1: Structures*, October 2004.

[45] W3C Recommendation. *XForms 1.0 (Second Edition)*, March 2006.

[46] The Workflow Management Coalition Specification. *Process Definition Interface – XML Process Definition Language*, October 2005.

[47] Zimmermann, O., Doubrovski, V., Grundler, J., and Hogg, K. Service-oriented architecture and business process choreography in an order management scenario: rationale, concepts, lessons learned. In *OOPSLA '05: Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications* (New York, NY, USA, 2005), ACM Press, pp. 301–312.

Appendix A

Review of IBM & SAP's Paper

The white paper [1] has been studied. Actors, use cases and classes have been identified.

Moreover scenario mentioned in this document have been adopted.

use case	figure
use case diagram	A.1
class diagram	A.2

A.1 Discussion

Actors and use cases have been identified for BPEL4People. Compared to the original propositions we propose the following additional roles:

Nominated Owner see 2.2.1

BPEL4People Administrator see 2.2.1

There are some requirements that we have paid special attention to as for example the four eyes scenario or the customisation of people links. Doing that we have added support for deadline parametrisation as well.

A few additional states have been added for tasks:

uninitialised A task is not ready before it's encapsulated people activity has been called, but in an uninitialised state.

escalated A task may be escalated when a deadline is not met.

nominated Nomination can take place for tasks.

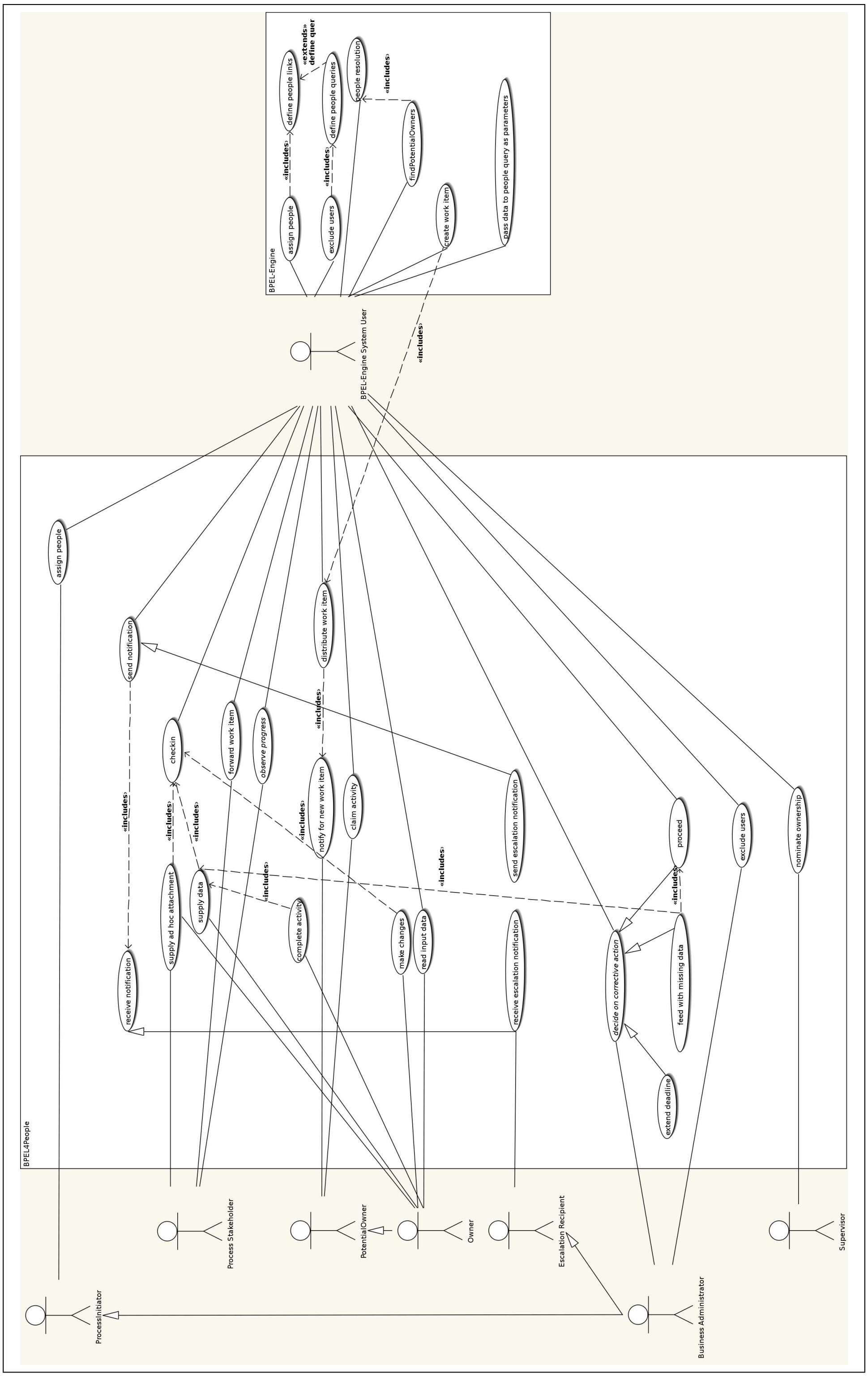

Figure A.1: BPEL4People Use Case Diagram

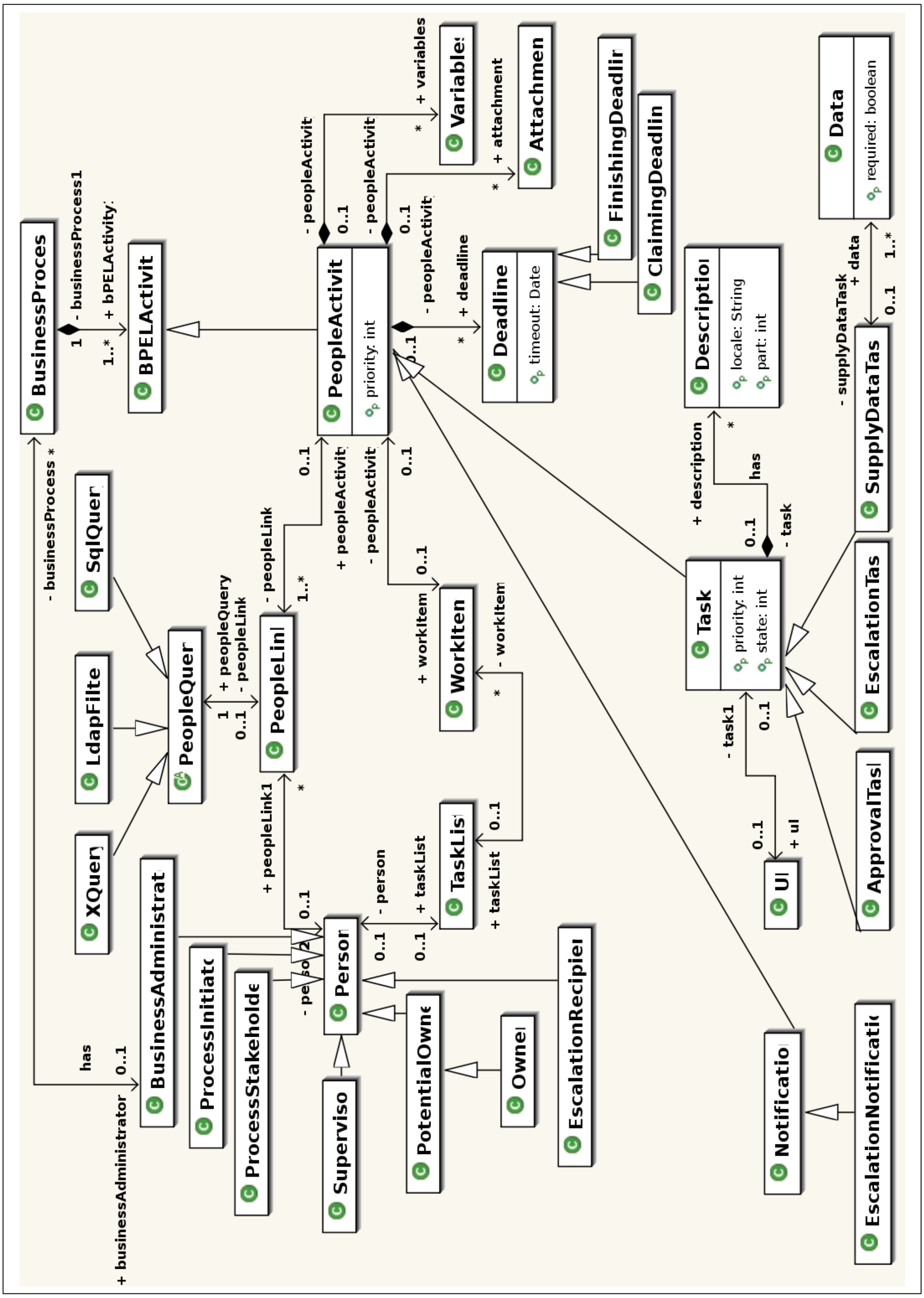

Figure A.2: BPEL4People Class Diagram

As we have decided to design a generic BPEL4People system we did not have to support the different models of interaction between tasks and processes as mentioned in section 4.3.4 of [1]. Within our system the models 1 - 3 exist and can be transformed into each other while model 5 represents the BPEL engine's point of view when calling a BPEL4People activity on our system.