

WS-BPEL Extension for People (BPEL4People) Specification Version 1.1

Committee Specification

17 August 2010

Specification URIs:

This Version:

http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cs-01.html http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cs-01.doc (Authoritative format) http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cs-01.pdf

Previous Version:

http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cd-09.html http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cd-09.doc (Authoritative format) http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cd-09.pdf

Latest Version:

http://docs.oasis-open.org/bpel4people/bpel4people-1.1.html http://docs.oasis-open.org/bpel4people/bpel4people-1.1.doc http://docs.oasis-open.org/bpel4people/bpel4people-1.1.pdf

Technical Committee:

OASIS BPEL4People TC

Chair:

Dave Ings, IBM

Editors:

Luc Clément, Active Endpoints, Inc. Dieter König, IBM Vinkesh Mehta, Deloitte Consulting LLP Ralf Mueller, Oracle Corporation Ravi Rangaswamy, Oracle Corporation Michael Rowley, Active Endpoints, Inc. Ivana Trickovic, SAP

Related work:

This specification is related to:

- BPEL4People WS-HumanTask Specification Version 1.1 http://docs.oasisopen.org/bpel4people/ws-humantask-1.1.html
- Web Services Business Process Execution Language Version 2.0 http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html

Declared XML Namespace:

b4p – http://docs.oasis-open.org/ns/bpel4people/bpel4people/200803

Abstract:

Web Services Business Process Execution Language, version 2.0 (WS-BPEL 2.0 or BPEL for brevity) introduces a model for business processes based on Web services. A BPEL process orchestrates interactions among different Web services. The language encompasses features needed to describe complex control flows, including error handling and compensation behavior. In practice, however many business process scenarios require human interactions. A process definition should incorporate people as another type of participants, because humans may also take part in business processes and can influence the process execution.

This specification introduces a BPEL extension to address human interactions in BPEL as a first-class citizen. It defines a new type of basic activity which uses human tasks as an implementation, and allows specifying tasks local to a process or use tasks defined outside of the process definition. This extension is based on the WS-HumanTask specification.

Status:

This document was last revised or approved by the OASIS WS-BPEL Extension for People Technical Committee on the above date. The level of approval is also listed above. Check the "Latest Version" or "Latest Approved Version" location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at http://www.oasis-open.org/committees/bpel4people/.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/bpel4people/ipr.php).

The non-normative errata page for this specification is located at http://www.oasis-open.org/committees/bpel4people/.

Notices

Copyright © OASIS® 2010. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see http://www.oasis-open.org/who/trademark.php for above guidance.

Table of Contents

1	Introduction	6
	1.1 Terminology	6
	1.2 Normative References	6
	1.3 Non-Normative References	7
	1.4 Conformance Targets	7
2	Language Design	9
	2.1 Dependencies on Other Specifications	9
	2.1.1 Namespaces Referenced	9
	2.2 Language Extensibility	9
	2.3 Overall Language Structure	9
	2.3.1 Syntax	10
	2.4 Default use of XPath 1.0 as an Expression Language	12
3	Concepts	13
	3.1 Generic Human Roles	13
	3.1.1 Syntax	13
	3.1.2 Initialization Behavior	. 14
	3.2 Assigning People	. 14
	3.2.1 Using Logical People Groups	. 14
	3.2.2 Computed Assignment	17
	3.3 Ad-hoc Attachments	17
4	People Activity	18
	4.1 Overall Syntax	19
	4.1.1 Properties	19
	4.2 Standard Overriding Elements	. 20
	4.3 People Activities Using Local Human Tasks	
	4.3.1 Syntax	21
	4.3.2 Examples	
	4.4 People Activities Using Local Notifications	. 22
	4.4.1 Syntax	. 23
	4.4.2 Examples	. 23
	4.5 People Activities Using Remote Human Tasks	
	4.5.1 Syntax	. 24
	4.5.2 Example	. 24
	4.5.3 Passing Endpoint References for Callbacks	. 24
	4.6 People Activities Using Remote Notifications	. 25
	4.6.1 Syntax	. 25
	4.6.2 Example	. 25
	4.7 Elements for Scheduled Actions	. 26
	4.8 People Activity Behavior and State Transitions	. 27
	4.9 Task Instance Data	. 28
	4.9.1 Presentation Data	. 28
	4.9.2 Context Data	. 29
	4.9.3 Operational Data	. 29

5	XPath Extension Functions	30
6	Coordinating Standalone Human Tasks	33
	6.1 Protocol Messages from the People Activity's Perspective	33
7	BPEL Abstract Processes	35
	7.1 Hiding Syntactic Elements	35
	7.1.1 Opaque Activities	35
	7.1.2 Opaque Expressions	35
	7.1.3 Opaque Attributes	35
	7.1.4 Opaque From-Spec	35
	7.1.5 Omission	
	7.2 Abstract Process Profile for Observable Behavior	35
	7.3 Abstract Process Profile for Templates	36
8	Conformance	37
Α.	. Standard Faults	38
В.	. Portability and Interoperability Considerations	39
C.	. BPEL4People Schema	40
D.	. Sample	46
	D.1 BPEL Definition	47
	D.2 WSDL Definitions	52
Ε.	. Acknowledgements	54
F.	•	

1 Introduction

- This specification introduces an extension to BPEL in order to support a broad range of scenarios that involve people within business processes.
- 4 The BPEL specification focuses on business processes the activities of which are assumed to be
- 5 interactions with Web services, without any further prerequisite behavior. But the spectrum of activities
- 6 that make up general purpose business processes is much broader. People often participate in the
- 7 execution of business processes introducing new aspects such as interaction between the process and
- 8 user interface, and taking into account human behavior. This specification introduces a set of elements
- 9 which extend the standard BPEL elements and enable the modeling of human interactions, which may
- 10 range from simple approvals to complex scenarios such as separation of duties, and interactions
- 11 involving ad-hoc data.

1

- 12 The specification introduces the people activity as a new type of basic activity which enables the
- specification of human interaction in processes in a more direct way. The implementation of a people
- 14 activity could be an inline task or a standalone human task defined in the WS-HumanTask specification
- 15 [WS-HumanTask]. The syntax and state diagram of the people activity and the coordination protocol that
- allows interacting with human tasks in a more integrated way is described. The specification also
- 17 introduces XPath extension functions needed to access the process context.
- The goal of this specification is to enable portability and interoperability:
- 19 Portability The ability to take design-time artifacts created in one vendor's environment and use them in
- another vendor's environment.
- 21 Interoperability The capability for multiple components (process infrastructure, task infrastructures and
- 22 task list clients) to interact using well-defined messages and protocols. This enables combining
- 23 components from different vendors allowing seamless execution.
- 24 Out of scope of this specification is how processes with human interactions are deployed or monitored.
- Usually people assignment is accomplished by performing queries on a people directory which has a
- certain organizational model. The mechanism of how an implementation evaluates people assignments,
- as well as the structure of the data in the people directory is also out of scope.

1.1 Terminology

- 29 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD
- 30 NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described
- 31 in RFC 2119 [RFC 2119].

1.2 Normative References

33 [BPEL4WS 1.1]

28

32

34

35

36 37

38

39

41

42 43

44

45

Business Process Execution Language for Web Services Version 1.1, BEA Systems, IBM, Microsoft, SAP AG and Siebel Systems, May 2003, available via http://www-

128.ibm.com/developerworks/library/specification/ws-bpel/

[RFC 2119]

Key words for use in RFCs to Indicate Requirement Levels, RFC 2119, available via http://www.ietf.org/rfc/rfc2119.txt

40 [RFC 3066]

Tags for the Identification of Languages, H. Alvestrand, IETF, January 2001, available via http://www.ietf.org/rfc/rfc3066.txt

[WS-Addr-Core]

Web Services Addressing 1.0 – Core, W3C Recommendation, May 2006, available via http://www.w3.org/TR/ws-addr-core

bpel4people-1.1-spec-cs-01 Copyright © OASIS® 2010. All Rights Reserved.

46 [WS-Addr-SOAP] 47 Web Services Addressing 1.0 - SOAP Binding, W3C Recommendation, May 2006, available via 48 http://www.w3.org/TR/ws-addr-soap 49 [WS-Addr-WSDL] 50 Web Services Addressing 1.0 – WSDL Binding, W3C Working Draft, February 2006, available via 51 http://www.w3.org/TR/ws-addr-wsdl 52 [WS-BPEL 2.0] 53 OASIS Standard, "Web Service Business Process Execution Language Version 2.0", 11 April 2007, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html 54 55 **IWSDL 1.11** Web Services Description Language (WSDL) Version 1.1, W3C Note, available via 56 57 http://www.w3.org/TR/2001/NOTE-wsdl-20010315 58 [WS-HumanTask] OASIS Committee Specification, "Web Services – Human Task (WS-HumanTask) 59 Specification Version 1.1, CS-01", 10 August 2010, http://docs.oasis-60 open.org/bpel4people/ws-humantask-1.1-spec-cs-01.doc 61 [XML Infoset] 62 63 XML Information Set, W3C Recommendation, available via http://www.w3.org/TR/2001/REC-xml-64 infoset-20011024/ 65 [XML Namespaces] 66 Namespaces in XML 1.0 (Second Edition), W3C Recommendation, available via 67 http://www.w3.org/TR/REC-xml-names/ 68 [XML Schema Part 1] 69 XML Schema Part 1: Structures, W3C Recommendation, October 2004, available via 70 http://www.w3.org/TR/xmlschema-1/ 71 [XML Schema Part 2] 72 XML Schema Part 2: Datatypes, W3C Recommendation, October 2004, available via 73 http://www.w3.org/TR/xmlschema-2/ 74 [XMLSpec] 75 XML Specification, W3C Recommendation, February 1998, available via http://www.w3.org/TR/1998/REC-xml-19980210 76 77 [XPATH 1.0] 78 XML Path Language (XPath) Version 1.0, W3C Recommendation, November 1999, available via http://www.w3.org/TR/1999/REC-xpath-19991116 79 1.3 Non-Normative References 80 81 There are no non-normative references made by this specification. 82

1.4 Conformance Targets

84

85

86

87

88

89

- As part of this specification, the following conformance targets are specified 83
 - **BPEL4People Definition** A BPEL4People Definition is a WS-BPEL 2.0 process definition that uses the BPEL4People extensions to WS-BPEL 2.0 specified in this document.
 - **BPEL4People Processor** A BPEL4People Processor is any implementation that accepts a BPEL4People definition and executes the semantics defined in this document.

90		
91		
92	•	WS-HumanTask Definition
93 94		A WS-HumanTask Definition is any artifact that complies with the human interaction schema and additional constraints as defined by the WS-HumanTask 1.1 specification.
95	•	WS-HumanTask Processor
96 97		A WS-HumanTask Processor is any implementation that accepts a WS-HumanTask definition and executes the semantics as defined by the WS-HumanTask 1.1 specification.

2 Language Design

- The BPEL4People extension is defined in a way that it is layered on top of BPEL so that its features can
- 100 be composed with BPEL features whenever needed. All elements and attributes introduced in this
- 101 extension are made available to both BPEL executable processes and abstract processes.
- This extension introduces a set of elements and attributes to cover different complex human interaction
- patterns, such as separation of duties, which are not defined as first-class elements.
- 104 Throughout this specification, WSDL and schema elements may be used for illustrative or convenience
- 105 purposes. However, in a situation where those elements or other text within this document contradict the
- 106 separate BPEL4People, WS-HumanTask, WSDL or schema files, it is those files that have precedence
- 107 and not this document.

98

108

116

117

126

135

2.1 Dependencies on Other Specifications

- 109 BPEL4People utilizes the following specifications:
- WS-BPEL 2.0: BPEL4People extends the WS-BPEL 2.0 process model and uses existing WS-BPEL 2.0 capabilities, such as those for data manipulation.
- WS-HumanTask 1.1: BPEL4People uses the definition of human tasks and, notifications, and extends generic human roles and people assignments introduced in WS-HumanTask 1.1.
- WSDL 1.1: BPEL4People uses WSDL for service interface definitions.
- XML Schema 1.0: BPEL4People utilizes XML Schema data model.
 - XPath 1.0: BPEL4People uses XPath as default query and expression language.

2.1.1 Namespaces Referenced

- 118 BPEL4People references these namespaces:
- htd http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803
- htt http://docs.oasis-open.org/ns/bpel4people/ws-humantask/types/200803
- bpel http://docs.oasis-open.org/wsbpel/2.0/process/executable
- abstract http://docs.oasis-open.org/wsbpel/2.0/process/abstract
- wsdl http://schemas.xmlsoap.org/wsdl/
- **xsd** http://www.w3.org/2001/XMLSchema
- xsi http://www.w3.org/2001/XMLSchema-instance

2.2 Language Extensibility

- 127 The BPEL4People specification extends the reach of the standard BPEL extensibility mechanism to
- 128 BPEL4People elements. This allows:
- 129 Attributes from other namespaces to appear on any BPEL4People element
- 130 Elements from other namespaces to appear within BPEL4People elements
- 131 Extension attributes and extension elements MUST NOT contradict the semantics of any attribute or
- 132 element from the BPEL4People namespace.
- 133 The standard BPEL element <extension> MUST be used to declare mandatory and optional
- 134 extensions of BPEL4People.

2.3 Overall Language Structure

- 136 This section explains the structure of BPEL4People extension elements, including the new activity type
- people activity, inline human tasks and people assignments.

2.3.1 Syntax

138139

140

Informal syntax of a BPEL process and scope containing logical people groups, inline human tasks, and people activity follows.

```
141
      <bpel:process b4p:shareComments="xsd:boolean"? ...</pre>
142
143
         xmlns:b4p="http://docs.oasis-open.org/ns/bpel4people/bpel4people/200803"
144
        xmlns:htd="http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803">
145
146
        <bpel:extensions>
147
           <bpel:extension</pre>
148
             namespace="http://docs.oasis-
149
      open.org/ns/bpel4people/bpel4people/200803"
150
             mustUnderstand="yes"/>
151
           <bpel:extension</pre>
152
             namespace="http://docs.oasis-open.org/ns/bpel4people/ws-
153
      humantask/200803"
154
             mustUnderstand="yes"/>
155
         </bre>
156
157
         <bpel:import</pre>
158
           importType="http://docs.oasis-open.org/ns/bpel4people/ws-
159
      humantask/200803" .../>
160
161
162
         <b4p:humanInteractions>?
163
164
           <htd:logicalPeopleGroups/>?
165
             <htd:logicalPeopleGroup name="NCName" reference="QName"?>+
166
167
             </htd:logicalPeopleGroup>
168
           </htd:logicalPeopleGroups>
169
170
           <htd:tasks>?
171
             <htd:task name="NCName">+
172
173
             </htd:task>
174
           </htd:tasks>
175
176
           <htd:notifications>?
177
             <htd:notification name="NCName">+
178
179
             </htd:notification>
180
           </htd:notifications>
181
182
         </bd></bd></ri>
183
184
         <br/><b4p:peopleAssignments>?
185
186
         </bd></bd>

</p
187
188
189
         <bpel:extensionActivity>
190
           <b4p:peopleActivity name="NCName" ...>
191
192
           </bd></bdp:peopleActivity>
193
         </bpel:extensionActivity>
194
```

- 195
- 196 A BPEL4People Definition MUST use BPEL4People extension elements and elements from WS-
- 197 HumanTask namespace. Therefore elements from namespaces BPEL4People and WS-HumanTask
- 198 MUST be understood.
- 199 The element <b4p:humanInteractions> is optional and contains declarations of elements from WS-
- 200 HumanTask namespace, that is <htd:logicalPeopleGroups>, <htd:tasks> and
- 201 <htd:notifications>.
- The element htd:logicalPeopleGroup specifies a logical people group used in an inline human
- task or a people activity. The name attribute specifies the name of the logical people group. The name
- 204 MUST be unique among the names of all logical people groups defined within the
- 205

b4p:humanInteractions> element.
- 206 The <htd:task> element is used to provide the definition of an inline human task. The syntax and
- semantics of the element are provided in the WS-HumanTask specification. The name attribute specifies
- the name of the task. The name MUST be unique among the names of all tasks defined within the
- 209 <htd:tasks> element.
- 210 The httd:notification> element is used to provide the definition of an inline notification. The syntax
- and semantics of the element are provided in the WS-HumanTask specification. The name attribute
- 212 specifies the name of the notification. The name MUST be unique among the names of all notifications
- 213 defined within the httd://doi.org/10.2016/j.june-10.2016/j.june-10.2016/httd://doi.org/10.2016/j.june-10.2016/httd://doi.org/10.2016/httd://doi.org/10.2016/httd://doi.org/10.2016/https://doi.org/10.2016/<a hre
- The element <b4p:peopleAssignments> is used to assign people to process-related generic human
- 215 roles. This element is optional. The syntax and semantics are introduced in section 3.1 "Generic Human
- 216 Roles".
- 217 New activity type <b4p:peopleActivity> is used to model human interactions within BPEL
- 219 used as wrapper. The syntax and semantics of the people activity are introduced in section 4 "People
- 220 Activity".

- 221 Any scope (or the process itself) can specify @b4p:shareComments="true" to specify that the
- comments that are added to any task executed within the scope (or a child scope) should be propagated
- 223 to any other task within the same scope that is started after the first task completes. When comments
- propagate to later tasks, all metadata for the comment MUST also be propagated.
- Note that, when a scope specifies the sharing of comments, it is not possible to override that sharing for
- 226 child or descendent scopes. When a scope specifies @b4p:shareComments="true" then child and
- 227 descendent scopes MUST NOT specify @b4p:shareComments="false". However, an individual
- 228 people activity can prevent its tasks' comments from being propagated by specifying
- 229 @dontShareComments="true".

```
231
```

257

258

259

260

261

262

263264

265 266

267

268

```
232
     <bpel:scope b4p:shareComments="xsd:boolean"? ...>
233
234
        <b4p:humanInteractions>?
235
236
        </bd></bd></b4p:humanInteractions>
237
238
        <bpel:extensionActivity>
239
          <b4p:peopleActivity name="NCName" dontShareComments="xsd:boolean" ...>
240
241
          </bd></bd>
242
        </bpel:extensionActivity>
243
244
      </bpel:scope>
```

BPEL scopes can also include elements from BPEL4People and WS-HumanTask namespaces except for the <b4p:peopleAssignments> element.

All BPEL4People Definition elements MAY use the element
b4p:documentation> to provide annotation for users. The content could be a plain text, HTML, and so on. The
b4p:documentation> element is optional and has the following syntax:

2.4 Default use of XPath 1.0 as an Expression Language

The XPath 1.0 specification [XPATH 1.0] defines the context in which an XPath expression is evaluated.
When XPath 1.0 is used as an Expression Language in BPEL4People or inlined WS-HumanTask language elements then the XPath context is initialized as follows:

Context node: none

Context position: none

Context size: none

- Variable bindings: all WS-BPEL variables visible to the enclosing element as defined by the WS-BPEL scope rules
- Function library: Core XPath 1.0, WS-BPEL, BPEL4People and WS-HumanTask functions MUST be available and processor-specific functions MAY be available
- Namespace declaration: all in-scope namespace declarations from the enclosing element

Note that XPath 1.0 explicitly requires that any element or attribute used in an XPath expression that does not have a namespace prefix must be treated as being namespace unqualified. As a result, even if there is a default namespace defined on the enclosing element, the default namespace will not be applied.

269 3 Concepts

Many of the concepts in BPEL4People are inherited from the WS-HumanTask specification so familiarity with this specification is assumed.

3.1 Generic Human Roles

Process-related generic human roles define what a person or a group of people resulting from a people assignment can do with the process instance. The process-related human roles complement the set of generic human roles specified in [WS-HumanTask]. There are three process-related generic human roles:

- Process initiator
- Process stakeholders
- Business administrators

Process initiator is the person associated with triggering the process instance at its creation time. The initiator is typically determined by the infrastructure automatically. This can be overridden by specifying a people assignment for process initiator. A BPEL4People Definition MAY define assignment for this generic human role. A compliant BPEL4People Processor MUST ensure that at runtime at least one person is associated with this role.

Process stakeholders are people who can influence the progress of a process instance, for example, by adding ad-hoc attachments, forwarding a task, or simply observing the progress of the process instance. The scope of a process stakeholder is broader than the actual BPEL4People specification outlines. The process stakeholder is associated with a process instance. If no process stakeholders are specified, the process initiator becomes the process stakeholder. A BPEL4People Definition MAY define assignment for this generic human role. A compliant BPEL4People Processor MUST ensure that at runtime at least one person is associated with this role.

Business administrators are people allowed to perform administrative actions on the business process, such as resolving missed deadlines. A business administrator, in contrast to a process stakeholder, has an interest in all process instances of a particular process type, and not just one. If no business administrators are specified, the process stakeholders become the business administrators. A BPEL4People Definition MAY define assignment for this generic human role. A compliant BPEL4People Processor MUST ensure that at runtime at least one person is associated with this role.

3.1.1 Syntax

The <code>genericHumanRole</code> abstract element introduced in the WS-HumanTask specification is extended with the following process-related human roles.

```
307
      <br/><b4p:peopleAssignments>?
308
309
        <br/><b4p:processInitiator>?
310
          <htd:from ...>...</htd:from>
311
        </bd></bdp:processInitiator>
312
313
        <b4p:processStakeholders>?
314
          <htd:from ...>...</htd:from>
315
        </bd></bde>
</bd>
316
```

322 Only process-related human roles MUST be used within the <b4p:peopleAssignments> element.

People are assigned to these roles as described in section 3.2 ("Assigning People").

3.1.2 Initialization Behavior

323

324

330 331

332

333

334

335

339

349

350

351

352

353 354

358

359

360

361

Assigning people to process-related generic human roles happens after BPEL process initialization (see [WS-BPEL 2.0], section 12.1). A BPEL4People Processor MUST initialize process-related generic human roles after the end of the initial start activity of the process and before processing other activities or links leaving the start activity. If that initialization fails then the fault b4p:initializationFailure MUST be thrown by a BPEL4People Processor.

3.2 Assigning People

To determine who is responsible for acting on a process, a human task or a notification in a certain generic human role, people need to be assigned. People assignment can be achieved in different ways:

- Via logical people groups (see 3.2.1 "Using Logical People Groups")
- Via literals (as introduced section 3.2.2 in [WS-HumanTask])
- Via expressions (see 3.2.2 "Computed Assignment")

When specifying people assignments then the data type htt:tOrganizationalEntity defined in [WS-HumanTask] is used. Using htt:tOrganizationalEntity allows to assign either a list of users or a list of unresolved groups of people ("work queues").

3.2.1 Using Logical People Groups

This section focuses on describing aspects of logical people groups that are specific to business processes. Logical people groups define which person or set of people can interact with a human task or a notification of a people activity. Details about how logical people groups are used with human tasks and notifications are provided by the WS-HumanTask specification.

Logical people groups can be specified as part of the business process definition. They can be defined either at the process level or on enclosed scopes. Definitions on inner scopes override definitions on outer scopes or the process respectively.

Logical people group definitions can be referenced by multiple people activities. Each logical people group is bound to a people query during deployment.

In the same way as in WS-HumanTask, a logical people group has one instance per set of unique arguments. Whenever a logical people group is referenced for the first time with a given set of unique arguments, a new instance MUST be created by the BPEL4People Processor. To achieve that, the logical people group MUST be evaluated / resolved for this set of arguments. Whenever a logical people group is referenced for which an in-stance already exists (i.e., it has already referenced before with the same set of arguments), the logical people group MAY be re-evaluated / re-resolved.

In particular, for a logical people group with no parameters, there is a single instance, which MUST be evaluated / resolved when the logical people group is first referenced, and which MAY be re-evaluated / re-resolved when referenced again.

Hence, using the same logical people group does not necessarily mean that the result of a people query is re-used, but that the same query is used to obtain a result. If the result of a previous people query needs to be re-used, then this result needs to be referenced explicitly from the process context. Please refer to section 5 "XPath Extension Functions" for a description of the syntax.

362 363

Assignment of Logical People Groups

A BPEL4People Definition MAY use the <assign> activity (see [WS-BPEL 2.0] section 8.4 for more details) to manipulate values of logical people group. A mechanism to assign to a logical people group or to assign from a logical people group using BPEL copy assignments is provided. The semantics of the <copy> activity introduced in [WS-BPEL 2.0] (see sections 8.4.1, 8.4.2 and 8.4.3 for more details) applies.

BPEL4People extends the from-spec and to-spec forms introduced in [WS-BPEL 2.0] as shown below:

In this form of from-spec and to-spec the <code>b4p:logicalPeopleGroup</code> attribute provides the name of a logical people group. The from-spec variant MAY include zero or more <code><b4p:argument></code> elements in order to pass values used in the people query. The <code>expressionLanguage</code> attribute specifies the language used in the expression. The attribute is optional. If not specified, the default language as inherited from the closest enclosing element that specifies the attribute is used.

Using a logical people group in the from-spec causes the evaluation of the logical people group. Logical people groups return data of type htt:torganizationalEntity. This data can be manipulated and assigned to other process variables using standard BPEL to-spec variable variants.

The new form of the from-spec can be used with the following to-spec variants:

• To copy to a variable

To copy to non-message variables and parts of message variables

```
<bpel:to expressionLanguage="anyURI"?>expression/bpel:to>
```

To copy to a property

```
<bpel:to variable="BPELVariableName" property="QName"/>
```

To copy to a logical people group

```
<bpel:to b4p:logicalPeopleGroup="NCName"/>
```

Using a logical people group in the to-spec of a <bpel:copy> assignment enables a set of people to be explicitly assigned. Whenever the logical people group is used after the assignment this assigned set of people is returned. Assigning values to a logical people group overrides what has been defined during deployment. This is true irrespective of any parameters specified for the logical people group.

The new form of the to-spec can be used with the following from-spec variants:

To copy from a variable

To copy from a property

```
<bpel:from variable="BPELVariableName" property="QName"/>
```

418

419

420

421

422 423

424

425 426

427

428

429

430 431 432

433 434 435

436 437

438 439

440 441

> 459 460 461

```
To copy from non-message variables and parts of message variables
```

<bpel:from expressionLanguage="anyURI"?>expression/bpel:from>

To copy from a literal value

```
<bpel:from>
 <bpel:literal>literal value
</bre>
```

To copy from a logical people group

```
<bpel:from b4p:logicalPeopleGroup="NCName"/>
```

Below are several examples illustrating the usage of logical people groups in copy assignments. The first example shows assigning the results of the evaluation of a logical people group to a process variable.

```
<bpel:assign name="getVoters">
  <bpel:copy>
    <bpel:from b4p:logicalPeopleGroup="voters">
      <br/><b4p:argument name="region">
        $electionRequest/region
      </bd></bd></ri>
    </break>
    <bpel:to variable="voters" />
  </bpel:copy>
</bpel:assign>
```

The next example demonstrates assigning a set of people to a logical people group using literal values.

```
<bpel:assign>
  <bpel:copy>
    <bpel:from>
      <bpel:literal>
        <htt:tOrganizationalEntity>
          <htt:user>Alan</htt:user>
          <htt:user>Dieter</htt:user>
          <htt:user>Frank</htt:user>
          <htt:user>Gerhard</htt:user>
          <htt:user>Ivana</htt:user>
          <htt:user>Karsten</htt:user>
          <htt:user>Matthias</htt:user>
          <htt:user>Patrick</htt:user>
        </htt:tOrganizationalEntity>
      </bpel:literal>
    </break>
    <bpel:to b4p:logicalPeopleGroup="bpel4peopleAuthors" />
  </bpel:copy>
</bpel:assign>
```

464

471

488

The third example shows assigning the results of one logical people group to another logical people group.

3.2.2 Computed Assignment

All computed assignment variants described in [WS-HumanTask] (see section 3.5 "Assigning People" for more details) are supported. In addition, the following variants are possible:

- The from-spec variant The from-spec variant The from variable is used to assign people that have been specified using a variable of the business process. The data type of the variable MUST be of type
- 481 htt:tOrganizationalEntity.
- 482 All other process context can be accessed using expressions of the following style:
- with XPath extension functions defined in section 5 "XPath Extension Functions". The
- 485 expressionLanguage attribute specifies the language used in the expression. The attribute is optional.
- 486 If not specified, the default language as inherited from the closest enclosing element that specifies the
- 487 attribute is used.

3.3 Ad-hoc Attachments

- Processes can have ad-hoc attachments. It is possible to exchange ad-hoc attachments between people
- 490 activities of a process by propagating ad-hoc attachments to and from the process level.
- When a people activity is activated, attachments from earlier tasks and from the process can be
- 492 propagated to its implementing human task. On completion of the human task, its ad-hoc attachments
- can be propagated to the process level, to make them globally available.
- 494 All manipulations of ad-hoc attachments at the process level are instantaneous, and not subject to
- 495 compensation or isolation.

4 People Activity

People activity is a basic activity used to integrate human interactions within BPEL processes. The following figure illustrates different ways in which human interactions (including human tasks and notifications) could be integrated.

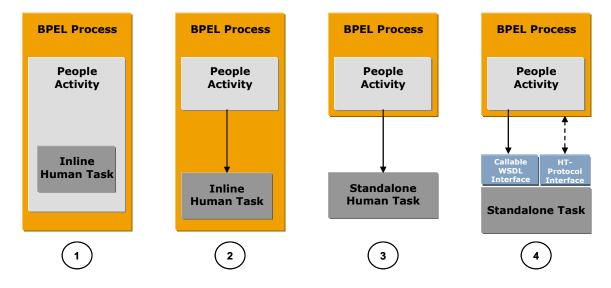


Figure 1: Constellations

Constellations 1 and 2 show models of interaction in which tasks are defined inline as part of a BPEL process. An *inline task* can be defined as part of a people activity (constellation 1). In this case, the use of the task is limited to the people activity encompassing it. Alternatively, a task can be defined as a top-level construct of the BPEL process or scope (constellation 2). In this case, the same task can be used within multiple people activities, which is significant from a reuse perspective. BPEL4People processes that use tasks in this way are portable among BPEL engines that implement BPEL4People. This also holds true for notifications.

Constellation 3 shows the use of a standalone task within the same environment, without the specification of a callable Web services interface on the task. Thus the task invocation is implementation-specific. This constellation is similar to constellation 2, except that the definition of the task is done independently of any process. As a result, the task has no direct access to process context. This also holds true for notifications.

Constellation 4 shows the use of a standalone task from a different environment. The major difference when compared to constellation 3 is that the task has a Web services callable interface, which is invoked using Web services protocols. In addition, the WS-HumanTask coordination protocol is used to communicate between processes and tasks (see section 6 "Coordinating Standalone Human Tasks" for more details on the WS-HumanTask coordination protocol). Using this mechanism, state changes are propagated between task and process activity, and the process can perform life cycle operations on the task, such as terminating it. BPEL4People processes that use tasks in this way are portable across different BPEL engines that implement BPEL4People. They are interoperable, assuming that both the process infrastructures and the task infrastructures implement the coordination protocol. In case of notifications a simplified protocol is used. For more detail on the relationship of WS-HumanTask and the BPEL4People specifications refer to section 1.1 of WS-HumanTask.

4.1 Overall Syntax

527 528

Definition of people activity:

```
529
       <bpel:extensionActivity>
530
531
         <b4p:peopleActivity name="NCName" inputVariable="NCName"?</pre>
532
            outputVariable="NCName"? isSkipable="xsd:boolean"?
533
            dontShareComments="xsd:boolean"?
534
            standard-attributes>
535
536
            standard-elements
537
538
            ( <htd:task>...</htd:task>
539
            | <b4p:localTask>...</b4p:localTask>
540
            | <b4p:remoteTask>...</b4p:remoteTask>
541
            | <htd:notification>...</htd:notification>
542
            | <b4p:localNotification>...</b4p:localNotification>
543
            | <b4p:remoteNotification>...</b4p:remoteNotification>
544
545
546
            <b4p:scheduledActions>? ...</b4p:scheduledActions>
547
548
            <bpel:toParts>?
549
              <bpel:toPart part="NCName" fromVariable="BPELVariableName" />+
550
            </bpel:toParts>
551
552
            <bpel:fromParts>?
553
              <bpel:fromPart part="NCName" toVariable="BPELVariableName" />+
554
            </bpel:fromParts>
555
556
            <b4p:attachmentPropagation fromProcess="all|none"</pre>
557
              toProcess="all|newOnly|none" />?
558
559
         </bd></bd>

</p
560
561
      </bpel:extensionActivity>
```

4.1.1 Properties

562

563

564 565

566

567

568 569

570

571 572

573574

575

576

577 578 The <b4p:peopleActivity> element is enclosed in the BPEL extensionActivity and has the following attributes and elements:

- outputVariable: This attribute refers to a process variable which is used as output of the WSDL operation of a task. The process variable in the BPEL4People Definition MUST have a WSDL message type. This attribute is optional. If the people activity uses a human task and this attribute is not present the Spel:fromParts> element MUST be used. The outputVariable attribute MUST NOT be used if the people activity uses a notification.
- isSkipable: This attribute indicates whether the task associated with the activity can be skipped at runtime or not. This is propagated to the task level. This attribute is optional. The default for this attribute is "no".
- dontShareComments: This attribute, if set to "true", indicates that comments that are added to the task associated with this people activity MUST NOT be propagated to any other task.

- standard-attributes: The activity makes available all BPEL's standard attributes.
 - standard-elements: The activity makes available all BPEL's standard elements.
 - o htd:task: This element is used to define an inline task within the people activity (constellation 1 in the figure above). This element is optional. Its syntax and semantics are introduced in section 4.3 "People Activities Using Local Human Tasks".
 - b4p:localTask: This element is used to refer to a standalone task with no callable Web service interface (constellations 2 or 3). This element is optional. Its syntax and semantics are introduced in section 4.3 "People Activities Using Local Human Tasks"
 - b4p:remoteTask: This element is used to refer to a standalone task offering callable Web service interface (constellation 4). This element is optional. Its syntax and semantics are introduced in section 4.5 "People Activities Using Remote Human Tasks".
 - htd:notification: This element is used to define an inline notification within the people activity (constellation 1 in the figure above). This element is optional. Its semantics is introduced in section 4.4 "People Activities Using Local Notifications".
 - o b4p:localNotification: This element is used to refer to a standalone notification with no callable Web service interface (constellations 2 or 3). This element is optional. Its semantics is introduced in section 4.4 "People Activities Using Local Notifications".
 - b4p:remoteNotification: This element is used to refer to a standalone notification offering callable Web service interface (constellation 4). This element is optional. Its syntax and semantics are introduced in section 4.6 "People Activities Using Remote Notifications".
 - b4p:scheduledActions: This element specifies when the task changes its state. Its syntax and semantics are introduced in section 4.7 "Elements for Scheduled Actions".
 - bpel:toParts: This element is used to explicitly create multi-part WSDL message from multiple BPEL variables. The element is optional. Its syntax and semantics are introduced in the WS-BPEL 2.0 specification, section 10.3.1. The

 inputVariable attribute are mutually exclusive.
 - bpel:fromParts: This element is used to assign values to multiple BPEL variables from an incoming multi-part WSDL message. The element is optional. Its syntax and semantics are introduced in the WS-BPEL 2.0 specification, section 10.3.1. The <bpel:fromParts> element and the outputVariable attribute are mutually exclusive. This element MUST NOT be used in a BPEL4People Definition if the people activity uses a notification.
 - b4p:attachmentPropagation: This element is used to describe the propagation behavior of ad-hoc attachments to and from the people activity. On activation of the people activity, either all ad-hoc attachments from the process are propagated to the people activity, so they become available to the corresponding task, or none. The fromProcess attribute is used to specify this. On completion of a people activity, all ad-hoc attachments are propagated to its process, or only newly created ones (but not those that were modified), or none. The toProcess attribute is used to specify this. The element is optional. The default value for this element is that all attachments are propagated from the process to the people activity and only new attachments are propagated back to the process.

4.2 Standard Overriding Elements

- Certain properties of human tasks and notifications can be specified on the process level as well as on local and remote task definitions and notification definitions allowing the process to override the original human task and notification definitions respectively. This increases the potential for reuse of tasks and notifications. Overriding takes place upon invocation of the Web service implemented by the human task (or notification) via the advanced interaction protocol implemented by both the process and the task (or notification).
- The following elements can be overridden:
- people assignments

628 • priority

 People assignments can be specified on remote and local human tasks and notifications. As a consequence, the invoked task receives the results of people queries performed by the business process on a per generic human role base. The result will be of type torganizationalEntity. The result needs to be understandable in the context of the task, i.e., the user identifiers and groups need to a) follow the same scheme and b) there exists a 1:1 relationship between the user identifiers and users. If a generic human role is specified on both the business process and the task it calls then the people assignment as determined by the process overrides what is specified on the task. In other words, the generic human roles defined at the task level provide the default. The same applies to people assignments on remote and local notifications.

The task's originator is set to the process stakeholder.

Priority of tasks and notifications can be specified on remote and local human tasks and notifications. If specified, it overrides the original priority of the human task (or notification).

Standard-overriding-elements is used in the syntax below as a shortened form of the following list of elements:

```
<htd:priority expressionLanguage="anyURI"? >
   integer-expression
</htd:priority>

<htd:peopleAssignments>?
   <htd:genericHumanRole>
        <htd:from>...</htd:from>
        </htd:genericHumanRole>
   </htd:peopleAssignments>
```

4.3 People Activities Using Local Human Tasks

People activities can be implemented using local human tasks. A local human task is one of the following:

- An inline task declared within the people activity. The task can be used only by that people activity
- An inline task declared within either the scope containing the people activity or the process scope. In this case the task can be reused as implementation of multiple people activities enclosed within the scope containing the task declaration
- A standalone task identified using a QName. In this case the task can be reused across multiple BPEL4People processes within the same environment.

The syntax and semantics of people activity using local tasks is given below.

4.3.1 Syntax

```
<b4p:peopleActivity inputVariable="NCName"? outputVariable="NCName"?</pre>
663
664
        isSkipable="xsd:boolean"? standard-attributes>
665
       standard-elements
666
667
        ( <htd:task>...</htd:task>
668
        | <b4p:localTask reference="QName">
669
           standard-overriding-elements
670
          </bd></bd></
671
672
673
     </bd></bd>
```

Properties

Element <htd:task> is used to define an inline task within the people activity. The syntax and semantics of the element are given in the WS-HumanTask specification. In addition, XPath expressions used in enclosed elements MAY refer to process variables. Enclosed elements MUST use the current value of the process variable. Changes to process variables MUST NOT directly cause changes in the execution of the enclosed elements, but only provide more current values when the enclosed elements choose to re-evaluate the expressions.

Element <b4p:localTask> is used to refer to a task enclosed in the BPEL4People process (a BPEL scope or the process scope) or a standalone task provided by the same environment. Attribute reference provides the QName of the task. The attribute is mandatory. The element MAY contain standard overriding elements explained in section 4.2 "Standard Overriding Elements".

4.3.2 Examples

682

683

684

685

686

687

708 709

710

711

712

713

714

715

716

717718

719

720 721

722

723

724

725

The following code shows a people activity declaring an inline task.

```
688
      <b4p:peopleActivity inputVariable="candidates"</pre>
689
                           outputVariable="vote"
690
                          isSkipable="yes">
691
        <htd:task>
692
          <htd:peopleAssignments>
693
            <htd:potentialOwners>
694
              <htd:from>$voters/users/user[i]</htd:from>
695
            </htd:potentialOwners>
696
          </htd:peopleAssignments>
        </htd:task>
697
698
        <bdd><bdp:scheduledActions></bd>
699
          <bd><bdp:expiration>
700
            <br/><b4p:documentation xml:lang="en-US">
701
              This people activity expires when not completed
702
              within 2 days after having been activated.
703
            </bd></bd></ri>
704
            <b4p:for>P2D</b4p:for>
705
          </bd></bd>
706
        </bd></bd></ri></ri>
707
     </bd></bd>
```

The following code shows a people activity referring to an inline task defined in the BPEL4People process.

```
<extensionActivity>
  <b4p:peopleActivity name="firstApproval"
    inputVariable="electionResult" outputVariable="decision">
        <b4p:localTask reference="tns:approveEmployeeOfTheMonth" />
        </b4p:peopleActivity>
  </extensionActivity>
```

4.4 People Activities Using Local Notifications

People activities can be implemented using local notifications. A local notification is one of the following:

- An inline notification declared within the people activity. The notification can be used only by that people activity
- An inline notification declared within either the scope containing the people activity or the process scope. In this case the notification can be reused as implementation of multiple people activities enclosed within the scope containing the notification declaration
- A standalone notification identified using a QName. In this case the notification can be reused across multiple BPEL4People processes within the same environment.

The syntax and semantics of people activity using local notifications is given below.

4.4.1 Syntax

```
728
     <b4p:peopleActivity name="NCName"? inputVariable="NCName"?</pre>
729
       standard-attributes>
730
       standard-elements
731
732
        ( <htd:notification>...</htd:notification>
733
       | <b4p:localNotification reference="QName">
734
           standard-overriding-elements
735
         </bd></bd></r>
736
737
     </bd></bd>
```

Properties

Element <htd:notification> is used to define an inline notification within the people activity. The syntax and semantics of the element are given in the WS-HumanTask specification. In addition, XPath expressions used in enclosed elements MAY refer to process variables. Enclosed elements MUST use the current value of the process variable. Changes to process variables MUST NOT directly cause changes in the execution of the enclosed elements, but only provide more current values when the enclosed elements choose to re-evaluate the expressions.

Element

Element

Element

4.4.2 Examples

The following code shows a people activity using a standalone notification.

4.5 People Activities Using Remote Human Tasks

People activities can be implemented using remote human tasks. This variant has been referred to as constellation 4 in Figure 1. The remote human task is invoked using a mechanism similar to the BPEL invoke activity: Partner link and operation identify the human task based Web service to be called. In addition to that, the name of a response operation on the *myRole* of the partner link is specified, allowing the human task based Web service to provide its result back to the calling business process.

Constellation 4 allows interoperability between BPEL4People compliant business processes of one vendor, and WS-HumanTask compliant human tasks of another vendor. For example, the communication to propagate state changes between the business process and the remote human task happens in a standardized way, as described in section 6 "Coordinating Standalone Human Tasks".

The remote human task can also define a priority element and people assignments. The priority and people assignments specified here override the original priority of the human task.

4.5.1 Syntax

The attribute responseOperation (of type xsd: NCName) specifies the name of the operation to be used to receive the response message from the remote human task. The operation attribute refers to an operation of the myRole port type of the partner link associated with the <b4p:remoteTask>. The attribute MUST be set in the BPEL4People Definition when the operation attribute refers to a WSDL one-way operation. The attribute MUST NOT be set when the operation attribute refers to a WSDL request-response operation.

4.5.2 Example

```
791
       <bpel:extensionActivity>
792
          <b4p:peopleActivity name="prepareInauguralSpeech"</pre>
793
                                     inputVariable="electionResult"
794
                                     outputVariable="speech"
795
                                     isSkipable="no">
796
             <b4p:remoteTask partnerLink="author"</pre>
797
                                  operation="prepareSpeech"
798
                                  responseOperation="receiveSpeech">
799
               <htd:priority>0</htd:priority> <!-- assign highest priority -->
800
               <htd:peopleAssignments>
801
                  <htd:potentialOwners>
802
                     <htd:from>$electionResult/winner</htd:from>
803
                  </htd:potentialOwners>
804
               </htd:peopleAssignments>
805
             </bd></bd>
806
          </bd></bd>

</p
807
       </bpel:extensionActivity>
```

4.5.3 Passing Endpoint References for Callbacks

A WS-HumanTask Processor MUST send a response message back to its calling process. The endpoint to which the response is to be returned to typically becomes known as late as when the human task is instantiated. This is no problem in case the human task is invoked synchronously via a request-response operation: a corresponding session between the calling process and the human task will exist and the response message of the human task uses this session.

But if the human task is called asynchronously via a one-way operation, such a session does not exist when the response message is sent. In this case, the BPEL4People Processor MUST pass the endpoint reference of the port expecting the response message of the human task to the WS-HumanTask Processor hosting the human task. Conceptually, this endpoint reference overrides any deployment settings for the human task. Besides the address of this port that endpoint reference MUST also specify additional metadata such that the port receiving the response is able to understand that the incoming message is in fact the response for an outstanding request (see [WS-HumanTask] section 8.2 for the definition of the metadata). Finally, such an endpoint reference MUST specify identifying data to allow the response message to be targeted to the correct instance of the calling process.

The additional metadata MAY consist of the name of the port type of the port as well as binding information about how to reach the port (see [WS-Addr-Core]) in order to support the replying activity of

the human task to send its response to the port. In addition, the name of the receiving operation at the calling process side is REQUIRED. This name MUST be provided as value of the responseOperation attribute of the <b4p:remoteTask> element (discussed in the previous section) and is passed together with an appropriate endpoint reference.

The above metadata represents the most generic solution allowing the response to be returned in all situations supported by WSDL. A simpler solution is supported in the case of the interaction between the calling process and the human task being based on SOAP: In this case, the metadata of the endpoint reference simply contains the value of the action header to be set in the response message.

In both cases (a request-response <b4p:remoteTask> as well as a <b4p:remoteTask> using two one-ways) the <b4p:remoteTask> activity is blocking. That is, the normal processing of a <b4p:remoteTask> activity does not end until a response message or fault message has been received from the human task. If the human task experiences a non-recoverable error, the WS-HumanTask Processor will signal that to the BPEL4People Processor and an b4p:nonRecoverableError fault MUST be raised in the parent process.

4.6 People Activities Using Remote Notifications

As described in the previous section, people activities can also be implemented using remote notifications. This variant is also referred to as *constellation 4*. Using remote notifications is very similar to using remote human tasks. Except for the name of the element enclosed in the people activity the main difference is that the remote notification is one-way by nature, and thus does not allow the specification of a response operation.

Remote notifications, like remote human tasks allow specifying properties that override the original properties of the notification Web service. The mechanism used is the same as described above. Like remote human tasks, remote notifications also allow overriding both people assignments and priority.

4.6.1 Syntax

829

830

831

832

833

834

835

836

837

838

839 840

841

842

843

844

845

846

847

848

856

4.6.2 Example

```
857
     <bpel:extensionActivity>
858
       <b4p:peopleActivity name="notifyEmployees"</pre>
859
                            inputVariable="electionResult">
860
          <b4p:remoteNotification partnerLink="employeeNotification"</pre>
861
                                  operation="receiveElectionResult">
862
           <htd:priority>5</htd:priority> <!-- assign moderate priority -->
863
           <htd:peopleAssignments>
864
             <htd:recipients>
865
                <htd:from>$voters</htd:from>
866
             </htd:recipients>
867
           </htd:peopleAssignments>
868
         </bd></bd>
869
       </bd></bd>

870
     </bpel:extensionActivity>
```

4.7 Elements for Scheduled Actions

Scheduled actions allow the specification of determining when a task needs to change its state. The following scheduled actions are defined:

DeferActivation: Specifies the activation time of the task. It is defined as either the period of time after which the task reaches state *Ready* (in case of explicit claim) or state *Reserved* (in case of implicit claim), or the point in time when the task reaches state *Ready* or state *Reserved*. The default value is zero, i.e. the task is immediately activated. If the activation time is defined as a point in time and the task is created after that point in time then the BPEL4People Processor MUST activate the task immediately.

Expiration: Specifies the expiration time of the task when the task becomes obsolete. It is defined as either the period of time after which the task expires or the point in time when the task expires. The time starts to be measured when the task enters state *Created*. If the task does not reach one of the final states (*Completed, Failed, Error, Exited, Obsolete*) by the expiration time the BPEL4People Processor MUST change the task state to *Exited*. Additional user-defined actions MUST NOT be performed. The default value is infinity, i.e. the task never expires. If the expiration time is defined as a point in time and the task is created after that point in time the BPEL4People Processor MUST change the task state to *Exited*. Note that deferred activation does not impact expiration. Therefore the task MAY expire even before being activated.

Element

b4p:scheduledActions> is used to include the definition of all scheduled actions within the task definition. If present, at least one scheduled activity MUST be defined in the BPEL4People Definition.

Syntax:

```
891
      <b4p:scheduledActions>?
892
893
        <bd><bdp:deferActivation>?</bd>
894
        ( <b4p:for expressionLanguage="anyURI"?>
895
            duration-expression
896
          </bd></bd></bd></br>
897
        | <b4p:until expressionLanguage="anyURI"?>
898
           deadline-expression
899
          </bde>
900
901
        </bd></deferActivation>
902
903
        <br/><b4p:expiration>?
904
        ( <b4p:for expressionLanguage="anyURI"?>
905
            duration-expression
906
          </ba>
907
        | <b4p:until expressionLanguage="anyURI"?>
908
            deadline-expression
909
          </ba></bdp:until>
910
911
        </bd></bde>
912
913
      </bd></bd></ri></base>
```

Properties

The <b4p:scheduledActions> element has the following optional elements:

- b4p:deferActivation: The element is used to specify activation time of the task. It includes the following elements:
 - b4p:for: The element is an expression which specifies the period of time (duration)
 after which the task reaches state Ready (in case of explicit claim) or state Reserved (in
 case of implicit claim). The absolute time of this transition is computed by adding the
 specified duration to the time at which the people activity begins execution.

o b4p:until: The element is an expression which specifies the point in time when the task reaches state *Ready* or state *Reserved*.

Elements <b4p:for> and <b4p:until> are mutually exclusive. There MUST be at least one <b4p:for> or <b4p:until> element.

- b4p:expiration: The element is used to specify the expiration time of the task when the task becomes obsolete:
 - b4p:for: The element is an expression which specifies the period of time (duration)
 after which the task expires. The absolute time of the expiration is computed by adding
 the duration to the time at which the people activity begins execution.
 - o b4p:until: The element is an expression which specifies the point in time when the task expires.

Elements <b4p:for> and <b4p:until> are mutually exclusive. There MUST be at least one <b4p:for> or <b4p:until> element.

The language used in expressions is specified using the <code>expressionLanguage</code> attribute. This attribute is optional. If not specified, the default language as inherited from the closest enclosing element that specifies the attribute is used.

If specified, the scheduledActions element MUST NOT be empty, that is one of the elements b4p:deferActivation and b4p:expiration MUST be defined.

Example:

923

924 925

926

927 928

929

930

931

932

933

934

935

936

937 938

939

940

941

961

962 963

```
942
      <b4p:scheduledActions>
943
944
        <br/><b4p:deferActivation>
945
          <b4p:documentation xml:lang="en-US">
946
            Activation of this task is deferred until the time specified
947
            in its input data.
948
          </bd></bd></
949
           <b4p:until>htd:getInput()/activateAt</b4p:until>
950
        </bd></deferActivation>
951
952
        <br/>b4p:expiration>
953
          <br/><b4p:documentation xml:lang="en-US">
954
            This task expires when not completed within 14 days after
955
            having been activated.
956
          </bd></bd></bdp:documentation>
957
          <b4p:for>P14D</b4p:for>
958
        </bd></bde>
959
960
      </bd></bd></ri></baseling</pre>
```

4.8 People Activity Behavior and State Transitions

Figure 2 shows the different states of the people activity and state transitions with associated triggers (events and conditions) and actions to be performed when transitions take place.

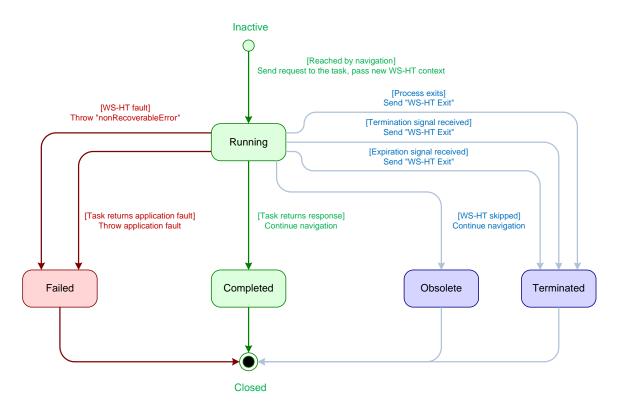


Figure 2: State diagram of the people activity

When the process execution instantiates a people activity this activity triggers the creation of a task in state *Running*. Upon receiving a response from the task, the people activity completes successfully and its state changes into the final state *Completed*.

If the task returns a fault, the people activity completes unsuccessfully and moves to final state *Failed* and the fault is thrown in the scope enclosing the people activity. If the task experiences a non-recoverable error, the people activity completes unsuccessfully and the standard fault nonRecoverableError is thrown in the enclosing scope.

The people activity goes to final state *Obsolete* if the task is skipped.

If the termination of the enclosed scope is triggered while the people activity is still running, the people activity is terminated prematurely and the associated running task is exited. A response for a terminated people activity MUST be ignored by the BPEL4People Processor.

If the task expires, the people activity is terminated prematurely and the associated task exits. In this case the standard fault b4p:taskExpired is thrown in the enclosing scope. When the process exits the people activity will also be terminated and the associated task is exited.

4.9 Task Instance Data

As defined by [WS-HumanTask], task instance data falls into the categories presentation data, context data, and operational data. Human tasks defined as part of a BPEL4People compliant business process have a superset of the instance data defined in [WS-HumanTask].

4.9.1 Presentation Data

The presentation data of tasks defined as part of a BPEL4People compliant business process is equivalent to that of a standalone human task.

4.9.2 Context Data

988

991

992

993

994

995 996

997

Tasks defined as part of a BPEL4People business process not only have access to the context data of the task, but also of the surrounding business process. The process context includes

- Process state like variables and ad-hoc attachments
- Values for all generic human roles of the business process, i.e. the process stakeholders, the business administrators of the process, and the process initiator
- Values for all generic human roles of human tasks running within the same business process

4.9.3 Operational Data

The operational data of tasks that is defined as part of a BPEL4People compliant business process is equivalent to that of a standalone human task.

5 XPath Extension Functions

This section introduces XPath extension functions that are provided to be used within the definition of a BPEL4People business process to access process context. Definition of these XPath extension functions is provided in the table below. Input parameters that specify peopleActivity name MUST be literal strings. This restriction does not apply to other parameters. Because XPath 1.0 functions do not support returning faults, an empty node set is returned in the event of an error.

Operation Name	Description	Parameters
getProcessStakeholders	Returns the stakeholders of the process. It MUST return an empty htt:organization alEntity in case of an error.	Out • organizational entity (htt:organizationalEnt ity)
getBusinessAdministrators	Returns the business administrators of the process. It MUST return an empty htt:organization alEntity in case of an error.	Out • organizational entity (htt:organizationalEnt ity)
getProcessInitiator	Returns the initiator of the process. It MUST return an empty htt:tUser in case of an error.	• the process initiator (htt:tUser)
getLogicalPeopleGroup	Returns the value of a logical people group. It MUST return an empty htt:organization alEntity in case of an error.	In • name of the logical people group (xsd:string) • The optional parameters that follow MUST appear in pairs. Each pair is defined as: • the qualified name of a logical people group parameter • the value for the named logical people group parameter; it can be an XPath expression Out • the value of the logical people group (htt:organizationalEnt)

Operation Name	Description	Parameters
		ity)
getActualOwner	Returns the actual owner of the task associated with the people activity. It MUST return an empty htt:tUser in case of an error.	In • people activity name (xsd:string) Out • the actual owner (htt:tUser)
getTaskInitiator	Returns the initiator of the task. Evaluates to an empty htt:user in case there is no initiator. It MUST return an empty htt:tUser in case of an error.	In • people activity name (xsd:string) Out • the task initiator (user id as htt:user)
getTaskStakeholders	Returns the stakeholders of the task. It MUST evaluate to an empty htt:organization alEntity in case of an error.	In • people activity name (xsd:string) Out • task stakeholders (htt:organizationalEnt ity)
getPotentialOwners	Returns the potential owners of the task associated with the people activity. It MUST return an empty htt:organization alEntity in case of an error.	In • people activity name (xsd:string) Out • potential owners (htt:organizationalEnt ity)
getAdministrators	Returns the administrators of the task associated with the people activity. It MUST return an empty htt:organization alEntity in case of an error.	In • people activity name (xsd:string) Out • business administrators (htt:organizationalEnt ity)
getTaskPriority	Returns the priority of the task associated with the people activity. It MUST evaluate to "5" in case the priority is not explicitly set.	<pre>In</pre>

Operation Name	Description	Parameters
getOutcome	Returns the task outcome of the task associated with the people activity	In • people activity name (xsd:string) Out • the task outcome (xsd:string)- if the outcome is not present, the empty nodeset MUST be returned.
getState	Returns the state of the people activity	In • people activity name (xsd:string) Out • the people activity state (xsd:string - see 4.8 People Activity Behavior and State Transitions)

1006 1007 XPath functions accessing data of a human task only guarantee to return data once the corresponding task has reached a final state.

6 Coordinating Standalone Human Tasks

Using the *WS-HT coordination protocol* introduced by [WS-HumanTask] (see section 7 "Interoperable Protocol for Advanced Interaction with Human Tasks" for more details) to control the autonomy and life cycle of human tasks, a BPEL process with a people activity can act as the parent application for remote human tasks.

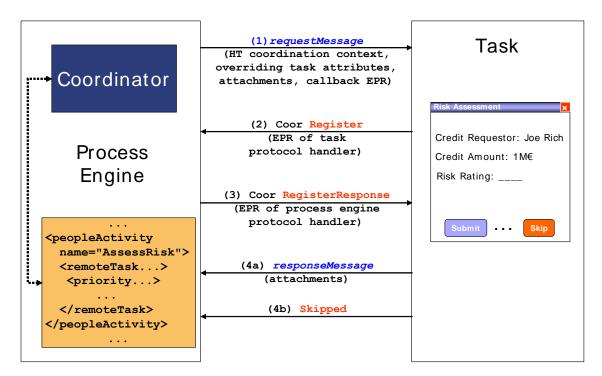


Figure 3: Message exchange between a people activity and a human task

Figure 3 shows some message exchanges between a BPEL process containing a people activity to perform a task (e.g. risk assessment) implemented by a remote human. The behavior of the people activity is the same as for a people activity with an inline human task. That behavior is achieved by coordinating the remote human task via the WS-HT coordination protocol.

6.1 Protocol Messages from the People Activity's Perspective

The BPEL4People Processor people activity MUST support the following behavior and the protocol messages exchanged with a standalone task. A summary is provided in the table below.

1. When the process execution reaches a people activity and determines that this activity can be executed, the BPEL4People Processor MUST create a WS-HT coordination context associated with the activity. This context is sent together with the request message to the appropriate service associated with the task. In addition, overriding attributes from the people activity, namely priority, people assignments, the skipable indicator and the task's expiration time, are sent. Also the BPEL4People Processor MAY propagate ad-hoc attachments from the process. All this information is sent as part of the header fields of the requesting message. These header fields as well as a corresponding mapping to SOAP headers are discussed in [WS-HumanTask].

When a response message is received from the task that indicates the successful completion of the task, the people activity completes. This response MAY include all new ad-hoc attachments from the human task.

1032

When a response message is received from the task that indicates a fault of the task, the people activity faults. The fault MUST be thrown in the scope of the people activity.

1035

4. When protocol message fault is received, the fault nonRecoverableError MUST be thrown in the scope enclosing the people activity.

1036

1037

5. When protocol message skipped is received, the people activity MUST move to state Obsolete.

1038 1039 6. If the task does not reach one of the final states by the expiration deadline, the people activity MUST be terminated. Protocol message exit is sent to the task.

1040

7. When the people activity is terminated, protocol message exit MUST be sent to the task.

10411042

8. When the process encounters an <exit> activity, protocol message exit MUST be sent to the task.

1042 1043 1044

The following table summarizes this behavior, the protocol messages sent, and their direction, i.e., whether a message is sent from the people activity to the task ("out" in the column titled Direction) or vice versa ("in").

10451046

Message	Direction	People activity behavior
application request with WS-HT coordination context (and callback information)	Out	People activity reached
task response	In	People activity completes
task fault response	In	People activity faults
Fault	In	People activity faults with b4p:nonRecoverableError
Skipped	In	People activity is set to obsolete
Exit	Out	Expired time-out
Exit	Out	People activity terminated
Exit	Out	<exit> encountered in enclosing process</exit>

7 BPEL Abstract Processes

- 1048 BPEL abstract processes are indicated by the namespace "http://docs.oasis-
- 1049 open.org/wsbpel/2.0/process/abstract". All constructs defined in BPEL4People extension
- 1050 namespaces MAY appear in abstract processes.

7.1 Hiding Syntactic Elements

- Opaque tokens defined in BPEL (activities, expressions, attributes and from-specs) MAY be used in
- 1053 BPEL4People extension constructs. The syntactic validity constraints of BPEL MUST apply in the same
- way to an Executable Completion of an abstract process containing BPEL4People extensions.

7.1.1 Opaque Activities

- BPEL4people does not change the way opaque activities can be replaced by an executable activity in an
- 1057 executable completion of an abstract process, that is, an <abstract:opaqueActivity> MAY also
- serve as a placeholder for a <bpel:extensionActivity> containing a <b4p:peopleActivity>.

7.1.2 Opaque Expressions

- Any expression introduced by BPEL4People MAY be made opaque. In particular, the following
- 1061 expressions MAY have the opaque="yes" attribute:
- 1062 <htd:argument name="NCName" expressionLanguage="anyURI"? opaque="yes" />
- 1063 <a href="
- 1065 <b4p:until expressionLanguage="anyURI"? opague="yes" />

1066 **7.1.3 Opaque Attributes**

- 1067 Any attribute introduced by BPEL4People MAY have an opaque value "##opaque" in an abstract
- 1068 process.

1069

1079

1051

1055

1059

7.1.4 Opaque From-Spec

- 1070 In BPEL, any from-spec in an executable process can be replaced by an opaque from-spec
- 1072 BPEL4People b4p:logicalPeopleGroup="NCName" attribute. In addition, the extension from-spec
- 1073 <htd:from> MAY also be replaced by an opaque from-spec in an abstract process.

1074 **7.1.5 Omission**

- 1075 In BPEL, omittable tokens are all attributes, activities, expressions and from-specs which are both (1)
- 1076 syntactically required by the Executable BPEL XML Schema, and (2) have no default value. This rule also
- 1077 applies to BPEL4People extensions in abstract processes. For example,

b4p:localTask
- 1078 reference="##opaque"> is equivalent to <b4p:localTask>.

7.2 Abstract Process Profile for Observable Behavior

- 1080 The Abstract Process Profile for Observable Behavior, indicated by the process attribute
- 1081 abstractProcessProfile="http://docs.oasis-
- open.org/wsbpel/2.0/process/abstract/ap11/2006/08", provides a means to create precise
- and predictable descriptions of observable behavior of the service(s) provided by an executable process.

The main application of this profile is the definition of business process contracts; that is, the behavior followed by one business partner in the context of Web services exchanges. A valid completion has to follow the same interactions as the abstract process, with the partners that are specified by the abstract process. The executable process can, however, perform additional interaction steps relating to other partners. Likewise, the executable process can perform additional human interactions. Beyond the restrictions defined in WS-BPEL 2.0, the use of opacity is not restricted in any way for elements and attributes introduced by BPEL4People.

7.3 Abstract Process Profile for Templates

- The Abstract Process Profile for Templates, indicated by the process attribute
- 1093 abstractProcessProfile="http://docs.oasis-
- open.org/wsbpel/2.0/process/abstract/simple-template/2006/08", allows the definition
- of Abstract Processes which hide almost any arbitrary execution details and have explicit opaque
- 1096 extension points for adding behavior.
- This profile does not allow the use of omission shortcuts but the use of opacity is not restricted in any
- 1098 way. For abstract processes belonging to this profile, this rule is extended to the elements and attributes
- 1099 introduced by BPEL4People.

1091

8 Conformance

The XML schema pointed to by the RDDL document at the namespace URI, defined by this specification, are considered to be authoritative and take precedence over the XML schema defined in the appendix of this document.

110311041105

1106

1107

1108 1109

1102

11001101

There are four conformance targets defined as part of this specification: a BPEL4People Definition, a BPEL4People Processor, a WS-HumanTask Definition and a WS-HumanTask Processor (see section 2.3). In order to claim conformance with BPEL4People 1.1, the conformance targets MUST comply with all normative statements in the BPEL4People and the WS-HumanTask specification, notably all MUST statements have to be implemented.

A. Standard Faults

1110

The following list specifies the standard faults defined within the BPEL4People specification. All standard fault names are qualified with the standard BPEL4People namespace.

Fault name	Description	
nonRecoverableError	Thrown if the task experiences a non-recoverable error.	
taskExpired	Thrown if the task expired.	

B. Portability and Interoperability Considerations

The following section illustrates the portability and interoperability aspects of the various usage constellations of BPEL4People with WS-HumanTask as described in Figure 1:

1116

1113

Portability - The ability to take design-time artifacts created in one vendor's environment and use them in another vendor's environment. Constellations one and two provide portability of BPEL4People processes with embedded human interactions in. Constellations three and four provide portability of BPEL4People processes with referenced human interactions.

1121

- Interoperability The capability for multiple components (process engine, task engine and task list client)
 to interact using well-defined messages and protocols. This enables to combine components from
 different vendors allowing seamless execution.
- 1125 Constellation four achieves interoperability between process and tasks from different vendor implementations.

1127

- 1128 Constellation 1
- Task definitions are defined inline of the people activities. Usage in this manner is typically for selfcontained people activities, whose tasks definitions are not intended to be reused elsewhere in the
 process or across multiple processes. This format will also provide scoping of the task definition since it
 will not be visible or accessible outside the people activity in which it is contained. Portability for this
 constellation requires support of both WS-HumanTask and BPEL4People artifacts using the inline task
 definition format. Since the process and task interactions are combined in one component, interoperability
 requirements are limited to those between the task list client and the infrastructure.

1136

- 1137 Constellation 2
- Similar to constellation 1, but tasks are defined at the process level. This allows task definitions to be referenced from within people activities enabling task reuse. Portability for this constellation requires support of both WS-HumanTask and BPEL4People artifacts using the process level scoped task definition format. Since the process and task interactions are combined in one component, interoperability requirements are limited to those between the task list client and the infrastructure.

1143

- 1144 Constellation 3
- In this constellation, the task and people activity definitions are defined as separate artifacts and execute in different infrastructure components but provided by the same vendor. Portability for this constellation requires support of both WS-HumanTask and BPEL4People as separate artifacts. Since the process and task components are implemented by the same vendor, interoperability requirements are limited to those between the task list client and the infrastructure.

- 1151 Constellation 4
- Identical to constellation 3 in terms of the task and people activity definitions, but in this case the process
 and task infrastructure are provided by different vendors. Portability for this constellation requires support
 of both WS-HumanTask and BPEL4People as separate artifacts. Interoperability between task and
- process infrastructures from different vendors is achieved using the WS-HumanTask coordination
- 1156 protocol.

1157 C. BPEL4People Schema

```
1158
      <?xml version="1.0" encoding="UTF-8"?>
1159
       <!--
1160
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1161
1162
       <xsd:schema</pre>
1163
         targetNamespace="http://docs.oasis-
1164
       open.org/ns/bpel4people/bpel4people/200803"
1165
         xmlns="http://docs.oasis-open.org/ns/bpel4people/bpel4people/200803"
1166
         xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
1167
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1168
         xmlns:htd="http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803"
1169
         elementFormDefault="qualified"
1170
         blockDefault="#all">
1171
1172
         <xsd:annotation>
1173
           <xsd:documentation>
1174
             XML Schema for BPEL4People 1.1 - WS-BPEL 2.0 Extension for Human Task
1175
       Interactions
1176
           </xsd:documentation>
         </xsd:annotation>
1177
1178
1179
        <!-- other namespaces -->
1180
         <xsd:import namespace="http://www.w3.org/XML/1998/namespace"</pre>
1181
           schemaLocation="http://www.w3.org/2001/xml.xsd" />
1182
         <xsd:import namespace="http://docs.oasis-open.org/ns/bpe14people/ws-</pre>
1183
       humantask/200803"
1184
           schemaLocation="ws-humantask.xsd" />
1185
         <xsd:import namespace="http://docs.oasis-</pre>
1186
       open.org/wsbpe1/2.0/process/executable"
1187
           schemaLocation="http://docs.oasis-
1188
       open.org/wsbpe1/2.0/OS/process/executable/ws-bpel executable.xsd" />
1189
1190
         <!-- base types for extensible elements -->
1191
         <xsd:complexType name="tExtensibleElements">
1192
           <xsd:sequence>
1193
             <xsd:element name="documentation" type="tDocumentation"</pre>
1194
               minOccurs="0" maxOccurs="unbounded" />
1195
             <xsd:any namespace="##other" processContents="lax" minOccurs="0"</pre>
1196
               maxOccurs="unbounded" />
1197
           </xsd:sequence>
1198
           <xsd:anyAttribute namespace="##other" processContents="lax" />
1199
         </xsd:complexType>
1200
         <xsd:complexType name="tExtensibleMixedNamespaceElements">
1201
           <xsd:sequence>
1202
             <xsd:element name="documentation" type="tDocumentation"</pre>
1203
               minOccurs="0" maxOccurs="unbounded" />
1204
             <xsd:element name="extensions" type="tExtensions" minOccurs="0" />
1205
           </xsd:sequence>
1206
           <xsd:anyAttribute namespace="##other" processContents="lax" />
1207
         </xsd:complexType>
1208
         <xsd:complexType name="tDocumentation" mixed="true">
1209
           <xsd:sequence>
1210
             <xsd:any namespace="##other" processContents="lax" minOccurs="0"</pre>
```

```
1211
               maxOccurs="unbounded" />
1212
           </xsd:sequence>
1213
           <xsd:attribute ref="xml:lang" />
1214
         </xsd:complexType>
1215
         <xsd:complexType name="tExtensions">
1216
           <xsd:sequence>
1217
             <xsd:any namespace="##other" processContents="lax" minOccurs="0"</pre>
1218
               maxOccurs="unbounded" />
1219
           </xsd:sequence>
1220
         </xsd:complexType>
1221
1222
         <!-- element "humanInteractions" to be used within "bpel:process" -->
1223
         <xsd:element name="humanInteractions" type="tHumanInteractions" />
1224
         <xsd:complexType name="tHumanInteractions">
1225
           <xsd:complexContent>
1226
             <xsd:extension base="tExtensibleMixedNamespaceElements">
1227
               <xsd:sequence>
1228
                 <xsd:element ref="htd:logicalPeopleGroups" minOccurs="0" />
1229
                 <xsd:element ref="htd:tasks" minOccurs="0" />
1230
                 <xsd:element ref="htd:notifications" minOccurs="0" />
1231
               </xsd:sequence>
1232
             </xsd:extension>
1233
           </xsd:complexContent>
1234
         </xsd:complexType>
1235
1236
        <!-- element "peopleAssignments" to be used within "bpel:process" -->
1237
        <xsd:element name="peopleAssignments" type="tPeopleAssignments" />
1238
         <xsd:complexType name="tPeopleAssignments">
1239
           <xsd:complexContent>
1240
             <xsd:extension base="tExtensibleElements">
1241
               <xsd:sequence>
1242
                 <xsd:element ref="genericHumanRole" minOccurs="1"</pre>
1243
      maxOccurs="unbounded" />
1244
               </xsd:sequence>
1245
             </xsd:extension>
1246
           </xsd:complexContent>
1247
         </xsd:complexType>
1248
1249
         <!-- element "genericHumanRole" within BPEL4People -->
1250
         <xsd:element name="genericHumanRole"</pre>
1251
       type="htd:tGenericHumanRoleAssignmentBase" abstract="true" block=""/>
1252
1253
         <xsd:element name="processStakeholders"</pre>
1254
       type="htd:tGenericHumanRoleAssignment" substitutionGroup="genericHumanRole"/>
1255
         <xsd:element name="businessAdministrators"</pre>
1256
       type="htd:tGenericHumanRoleAssignment" substitutionGroup="genericHumanRole"/>
1257
         <xsd:element name="processInitiator" type="htd:tGenericHumanRoleAssignment"</pre>
1258
       substitutionGroup="genericHumanRole"/>
1259
1260
         <!-- element "argument" to be used within "bpel:from" -->
1261
         <xsd:element name="argument" type="tArgument" />
1262
         <xsd:complexType name="tArgument">
1263
           <xsd:complexContent>
1264
             <xsd:extension base="bpel:tExpression">
1265
               <xsd:attribute name="name" type="xsd:NCName" />
1266
             </xsd:extension>
1267
           </xsd:complexContent>
1268
         </xsd:complexType>
```

```
1269
1270
         <!-- attribute "logicalPeopleGroup" to be used within "bpel:from" and
1271
1272
         <xsd:attribute name="logicalPeopleGroup" type="xsd:NCName" />
1273
1274
         <!-- attribute "shareComments" to be used within "bpel:process" and
1275
       "bpel:scope" -->
1276
         <xsd:attribute name="shareComments" type="xsd:boolean" />
1277
1278
         <!-- element "peopleActivity" to be used within "bpel:extensionActivity" --
1279
1280
        <xsd:element name="peopleActivity" type="tPeopleActivity" />
1281
         <xsd:complexType name="tPeopleActivity">
1282
           <xsd:complexContent>
1283
             <xsd:extension base="tExtensibleMixedNamespaceElements">
1284
               <xsd:sequence>
1285
                 <xsd:element ref="bpel:targets" minOccurs="0" />
1286
                 <xsd:element ref="bpel:sources" minOccurs="0" />
1287
                 <xsd:choice>
1288
                   <xsd:element ref="htd:task" />
1289
                   <xsd:element ref="localTask" />
1290
                   <xsd:element ref="remoteTask" />
1291
                   <xsd:element ref="htd:notification" />
1292
                   <xsd:element ref="localNotification" />
1293
                   <xsd:element ref="remoteNotification" />
1294
                 </xsd:choice>
1295
                 <xsd:element ref="scheduledActions" minOccurs="0" />
                 <xsd:element ref="toParts" minOccurs="0" />
1296
1297
                 <xsd:element ref="fromParts" minOccurs="0" />
1298
                 <xsd:element ref="attachmentPropagation" minOccurs="0" />
1299
                 <xsd:any namespace="##other" processContents="lax"</pre>
1300
                   minOccurs="0" maxOccurs="unbounded" />
1301
              </xsd:sequence>
1302
               <xsd:attribute name="name" type="xsd:NCName" />
1303
               <xsd:attribute name="suppressJoinFailure" type="tBoolean"</pre>
1304
                use="optional" />
1305
               <xsd:attribute name="inputVariable" type="xsd:QName" />
1306
              <xsd:attribute name="outputVariable" type="xsd:QName" />
1307
              <xsd:attribute name="isSkipable" type="tBoolean"</pre>
1308
                use="optional" default="no" />
1309
               <xsd:attribute name="dontShareComments" type="tBoolean"</pre>
1310
                 use="optional" default="no" />
1311
             </xsd:extension>
1312
           </xsd:complexContent>
1313
         </xsd:complexType>
         <xsd:complexType name="t0verridableTaskElements">
1314
1315
           <xsd:complexContent>
1316
             <xsd:extension base="tExtensibleMixedNamespaceElements">
1317
               <xsd:sequence>
1318
                 <xsd:element ref="htd:priority" minOccurs="0" />
1319
                 <xsd:element ref="htd:peopleAssignments" minOccurs="0" />
1320
               </xsd:sequence>
1321
             </xsd:extension>
1322
           </xsd:complexContent>
1323
         </xsd:complexType>
1324
         <xsd:element name="localTask" type="tLocalTask" />
1325
         <xsd:complexType name="tLocalTask">
1326
           <xsd:complexContent>
```

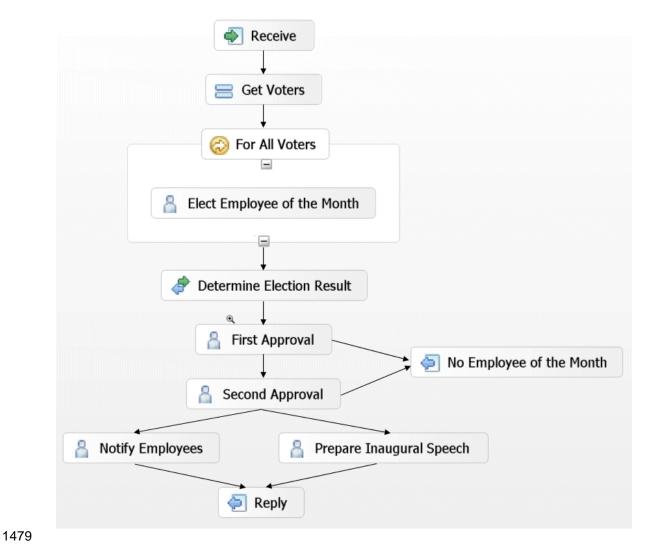
```
1327
             <xsd:extension base="tOverridableTaskElements">
1328
               <xsd:attribute name="reference" type="xsd:QName"</pre>
1329
                 use="required" />
1330
             </xsd:extension>
1331
           </xsd:complexContent>
1332
         </xsd:complexType>
1333
         <xsd:element name="remoteTask" type="tRemoteTask" />
1334
         <xsd:complexType name="tRemoteTask">
1335
           <xsd:complexContent>
1336
             <xsd:extension base="tOverridableTaskElements">
               <xsd:attribute name="partnerLink" type="xsd:NCName"</pre>
1337
1338
                 use="required" />
1339
               <xsd:attribute name="operation" type="xsd:NCName"</pre>
1340
                 use="required" />
1341
               <xsd:attribute name="responseOperation" type="xsd:NCName" />
1342
             </xsd:extension>
1343
           </xsd:complexContent>
1344
         </xsd:complexType>
1345
         <xsd:complexType name="tOverridableNotificationElements">
1346
           <xsd:complexContent>
1347
             <xsd:extension base="tExtensibleMixedNamespaceElements">
1348
               <xsd:sequence>
1349
                 <xsd:element ref="htd:priority" minOccurs="0" />
1350
                 <xsd:element ref="htd:peopleAssignments" minOccurs="0" />
1351
               </xsd:sequence>
1352
             </xsd:extension>
1353
           </xsd:complexContent>
1354
         </xsd:complexType>
1355
         <xsd:element name="localNotification" type="tLocalNotification" />
1356
         <xsd:complexType name="tLocalNotification">
1357
           <xsd:complexContent>
1358
             <xsd:extension base="t0verridableNotificationElements">
1359
               <xsd:attribute name="reference" type="xsd:QName"</pre>
1360
                 use="required" />
1361
             </xsd:extension>
1362
           </xsd:complexContent>
1363
         </xsd:complexType>
1364
         <xsd:element name="remoteNotification" type="tRemoteNotification" />
1365
         <xsd:complexType name="tRemoteNotification">
1366
           <xsd:complexContent>
             <xsd:extension base="t0verridableNotificationElements">
1367
1368
               <xsd:attribute name="partnerLink" type="xsd:NCName"</pre>
1369
                 use="required" />
1370
               <xsd:attribute name="operation" type="xsd:NCName"</pre>
1371
                 use="required" />
1372
             </xsd:extension>
1373
           </xsd:complexContent>
1374
         </xsd:complexType>
1375
         <xsd:element name="scheduledActions" type="tScheduledActions" />
1376
         <xsd:complexType name="tScheduledActions">
1377
           <xsd:complexContent>
             <xsd:extension base="tExtensibleElements">
1378
1379
               <xsd:sequence>
1380
                 <xsd:element name="deferActivation"</pre>
1381
                   type="tScheduledActionsDetails" minOccurs="0" />
1382
                 <xsd:element name="expiration"</pre>
1383
                   type="tScheduledActionsDetails" minOccurs="0" />
1384
               </xsd:sequence>
```

```
1385
             </xsd:extension>
1386
           </xsd:complexContent>
1387
         </xsd:complexType>
1388
         <xsd:complexType name="tScheduledActionsDetails">
1389
           <xsd:complexContent>
1390
             <xsd:extension base="tExtensibleElements">
1391
               <xsd:sequence>
1392
                 <xsd:choice>
1393
                   <xsd:element name="for" type="bpel:tDuration-expr" />
1394
                   <xsd:element name="until" type="bpel:tDeadline-expr" />
1395
                 </xsd:choice>
1396
               </xsd:sequence>
1397
             </xsd:extension>
1398
           </xsd:complexContent>
1399
         </xsd:complexType>
1400
         <xsd:element name="fromParts" type="tFromParts" />
1401
         <xsd:complexType name="tFromParts">
1402
           <xsd:complexContent>
1403
             <xsd:extension base="tExtensibleElements">
1404
               <xsd:sequence>
1405
                 <xsd:element ref="fromPart" maxOccurs="unbounded" />
1406
               </xsd:sequence>
1407
             </xsd:extension>
1408
           </xsd:complexContent>
1409
         </xsd:complexType>
1410
         <xsd:element name="fromPart" type="tFromPart" />
1411
         <xsd:complexType name="tFromPart">
1412
           <xsd:complexContent>
1413
             <xsd:extension base="tExtensibleElements">
1414
               <xsd:attribute name="part" type="xsd:NCName" use="required" />
1415
               <xsd:attribute name="toVariable" type="bpel:BPELVariableName"</pre>
1416
                 use="required" />
1417
             </xsd:extension>
1418
           </xsd:complexContent>
1419
         </xsd:complexType>
1420
         <xsd:element name="toParts" type="tToParts" />
1421
         <xsd:complexType name="tToParts">
1422
           <xsd:complexContent>
             <xsd:extension base="tExtensibleElements">
1423
1424
               <xsd:sequence>
1425
                 <xsd:element ref="toPart" maxOccurs="unbounded" />
1426
               </xsd:sequence>
1427
             </xsd:extension>
1428
           </xsd:complexContent>
1429
         </xsd:complexType>
1430
         <xsd:element name="toPart" type="tToPart" />
1431
         <xsd:complexType name="tToPart">
1432
           <xsd:complexContent>
1433
             <xsd:extension base="tExtensibleElements">
1434
               <xsd:attribute name="part" type="xsd:NCName" use="required" />
1435
               <xsd:attribute name="fromVariable"</pre>
1436
                 type="bpel:BPELVariableName" use="required" />
1437
             </xsd:extension>
1438
           </xsd:complexContent>
1439
         </xsd:complexType>
1440
         <xsd:element name="attachmentPropagation"</pre>
1441
           type="tAttachmentPropagation" />
1442
         <xsd:complexType name="tAttachmentPropagation">
```

```
1443
           <xsd:complexContent>
1444
             <xsd:extension base="tExtensibleElements">
1445
               <xsd:attribute name="fromProcess" type="tFromProcess"</pre>
1446
                 default="all" />
1447
               <xsd:attribute name="toProcess" type="tToProcess"</pre>
1448
                 default="newOnly" />
1449
             </xsd:extension>
1450
           </xsd:complexContent>
1451
         </xsd:complexType>
1452
         <xsd:simpleType name="tFromProcess">
1453
           <xsd:restriction base="xsd:string">
1454
             <xsd:enumeration value="all" />
1455
             <xsd:enumeration value="none" />
1456
           </xsd:restriction>
1457
         </xsd:simpleType>
1458
         <xsd:simpleType name="tToProcess">
1459
           <xsd:restriction base="xsd:string">
1460
             <xsd:enumeration value="all" />
1461
             <xsd:enumeration value="newOnly" />
1462
             <xsd:enumeration value="none" />
1463
           </xsd:restriction>
1464
         </xsd:simpleType>
1465
1466
         <!-- miscellaneous helper elements and types -->
1467
         <xsd:simpleType name="tBoolean">
1468
           <xsd:restriction base="xsd:string">
1469
             <xsd:enumeration value="yes" />
1470
             <xsd:enumeration value="no" />
1471
           </xsd:restriction>
1472
         </xsd:simpleType>
1473
1474
      </xsd:schema>
```

D. Sample

This appendix contains a sample that outlines the basic concepts of this specification. The sample process implements the election of the "Employee of the month" in a fictious company. The structure of the business process is shown in the figure below:



The process is started and as a first step, the people are determined that qualify as voters for the "Employee of the month". Next, all the voters identified before get a chance to cast their votes. After that, the election result is determined by counting the votes casted. After the result is clear, two different people from the set of people entitled to approve the election either accept or reject the voting result. In case any of the two rejects, then there is no "Employee of the month" elected in the given month, and the process ends. In case all approvals are obtained successfully, the employees are notified about the outcome of the election, and a to-do is created for the elected "Employee of the month" to prepare an inaugural speech. Once this is completed, the process completes successfully.

The sections below show the definition of the BPEL process implementing the "Employee of the month" process.

D.1 BPEL Definition

```
1491
      <?xml version="1.0" encoding="UTF-8"?>
1492
1493
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1494
1495
      cprocess name="EmployeeOfTheMonthProcess"
        targetNamespace="http://www.example.com"
1496
1497
         xmlns:tns="http://www.example.com"
1498
         xmlns:hr="http://www.example.com/approval"
1499
         xmlns:el="http://www.example.com/election"
        xmlns:ty="http://www.example.com/types"
1500
1501
        xmlns:ta="http://www.example.com/tasks"
1502
        xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
1503
        xmlns:b4p="http://docs.oasis-open.org/ns/bpel4people/bpel4people/200803"
1504
        xmlns:htd="http://docs.oasis-open.org/ns/bpe14people/ws-humantask/200803"
1505
         xmlns:htt="http://docs.oasis-open.org/ns/bpel4people/ws-
1506
      humantask/types/200803"
1507
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1508
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1509
        xsi:schemaLocation="http://docs.oasis-
      open.org/ns/bpel4people/bpel4people/200803 ../../xml/bpel4people.xsd
1510
1511
      http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803 ../../xml/ws-
1512
       humantask.xsd http://docs.oasis-open.org/ns/bpel4people/ws-
1513
      humantask/types/200803 ../../xml/ws-humantask-types.xsd">
1514
1515
         <documentation>
1516
           Example for BPEL4People 1.1 - WS-BPEL 2.0 Process with BPEL4People
1517
      Extensions
1518
        </documentation>
1519
1520
         <br/><b4p:humanInteractions>
1521
1522
           <htd:logicalPeopleGroups>
1523
1524
             <htd:logicalPeopleGroup name="voters">
1525
               <htd:documentation xml:lang="en-US">
1526
                 The group entitled to vote the employee of the month for the
1527
                 given region.
1528
               </htd:documentation>
1529
               <htd:parameter name="region" type="xsd:string" />
1530
             </htd:logicalPeopleGroup>
1531
1532
             <htd:logicalPeopleGroup name="approvers">
1533
               <htd:documentation xml:lang="en-US">
1534
                 The group entitled to approve the elected employee of the
1535
                 month for the given region.
1536
               </htd:documentation>
1537
               <htd:parameter name="region" type="xsd:string" />
1538
             </htd:logicalPeopleGroup>
1539
1540
             <htd:logicalPeopleGroup name="employees">
1541
               <htd:documentation xml:lang="en-US">
1542
                 The group of employees to be notified about the election
1543
                 result of the employee of the month election for the given
1544
                 region.
1545
               </htd:documentation>
1546
               <htd:parameter name="region" type="xsd:string" />
```

```
1547
             </htd:logicalPeopleGroup>
1548
1549
             <htd:logicalPeopleGroup name="regionalElectionCommittee">
1550
               <htd:documentation xml:lang="en-US">
1551
                 The group who is in charge for the election of the
1552
                 employee of the month election for the given region.
1553
               </htd:documentation>
               <htd:parameter name="region" type="xsd:string" />
1554
1555
             </htd:logicalPeopleGroup>
1556
1557
           </htd:logicalPeopleGroups>
1558
1559
           <htd:tasks>
1560
             <htd:task name="approveEmployeeOfTheMonth">
1561
               <htd:documentation xml:lang="en-US">
1562
                  The reusable definition of the task used to approve the
1563
                  election of the employee of the month.
1564
               </htd:documentation>
1565
               <htd:interface operation="approve" portType="hr:approvalPT"/>
1566
               <htd:peopleAssignments>
1567
                 <htd:potentialOwners>
1568
                   <htd:from logicalPeopleGroup="approvers">
1569
                     <!-- variables used here need to be defined on the
1570
                          enclosing scope or above -->
1571
                     <htd:argument name="region">
1572
                       $electionRequest/region
1573
                     </htd:argument>
1574
                   </htd:from>
1575
                 </htd:potentialOwners>
1576
               </htd:peopleAssignments>
1577
               <htd:presentationElements/>
1578
             </htd:task>
1579
           </htd:tasks>
1580
         </bd></bd></
1581
1582
1583
         <br/><b4p:peopleAssignments>
1584
1585
           <bd/>
<bd><bd>processStakeholders></bd>
1586
             <htd:from logicalPeopleGroup="regionalElectionCommittee">
1587
               <htd:argument name="region">
1588
                 $electionRequest/region
1589
               </htd:argument>
1590
             </htd:from>
           </ba>
1591
1592
           <b4p:businessAdministrators>
1593
1594
             <htd:from>
1595
               <htd:literal>
1596
                 <htt:organizationalEntity>
1597
                   <htt:user>Peter</htt:user>
1598
                   <htt:user>Paul</htt:user>
1599
                   <htt:user>Mary</htt:user>
1600
                 </htt:organizationalEntity>
1601
               </htd:literal>
1602
             </htd:from>
1603
           </bd></bdp:businessAdministrators>
1604
```

```
1605
          </bd></bd>

</p
1606
1607
          <extensions>
1608
            <extension
1609
              namespace="http://docs.oasis-
1610
       open.org/ns/bpel4people/bpel4people/200803"
1611
              mustUnderstand="yes"/>
1612
            <extension
1613
              namespace="http://docs.oasis-open.org/ns/bpel4people/ws-
1614
       humantask/200803"
1615
              mustUnderstand="yes"/>
1616
          </extensions>
1617
1618
          <import</pre>
1619
            importType="http://www.w3.org/2001/XMLSchema"
1620
            namespace="http://www.example.com/types"/>
1621
          <import</pre>
1622
            importType="http://www.example.org/WS-HT"
1623
            namespace="http://www.example.com/tasks"/>
1624
1625
            importType="http://schemas.xmlsoap.org/wsdl/"
1626
            namespace="http://www.example.com/election"
1627
            location="bpel4people-example-election.wsdl"/>
1628
          <import</pre>
1629
            importType="http://schemas.xmlsoap.org/wsdl/"
1630
            namespace="http://www.example.com/approval"
1631
            location="bpel4people-example-approval.wsdl"/>
1632
1633
          <partnerLinks>
1634
            <partnerLink partnerLinkType="electionPLT"</pre>
1635
                           name="electionPL"/>
1636
          </partnerLinks>
1637
1638
          <variables>
1639
            <variable name="candidates" type="htt:users"/>
1640
            <variable name="voters" type="htd:tOrganizationalEntity"/>
1641
            <variable name="electionRequest" type="ty:electionRequestData"/>
            <variable name="electionResult" type="ty:electionResultData"/>
1642
1643
            <variable name="decision" type="xsd:boolean"/>
1644
            <variable name="speech" type="ty:document"/>
1645
          </variables>
1646
1647
          <sequence>
1648
1649
            <receive partnerLink="electionPL"</pre>
1650
                      portType="el:electionPT"
1651
                      operation="elect"
1652
                      variable="electionRequest"
1653
                      createInstance="yes"/>
1654
1655
            <assign name="getVoters">
1656
              <copy>
1657
                <from>$electionRequests/candidates</from>
                 <to variable="candidates"/>
1658
1659
              </copy>
1660
              <copy>
1661
                 <from b4p:logicalPeopleGroup="voters">
1662
                   <br/><b4p:argument name="region">
```

```
1663
                   $electionRequest/region
1664
                 </bar></bareargument>
1665
               </from>
1666
               <to variable="voters" />
1667
             </copy>
1668
           </assign>
1669
1670
           <forEach counterName="i" parallel="yes">
1671
             <startCounterValue> 1 </startCounterValue>
1672
             <finalCounterValue>
1673
               count($voters/users/user)
1674
             </finalCounterValue>
1675
1676
             <scope>
1677
               <variables>
1678
                  <variable name="vote" type="htt:user"/>
1679
               </variables>
1680
1681
               <sequence>
1682
                 <!-- Constellation 1 -->
1683
                 <extensionActivity>
1684
                   <b4p:peopleActivity name="electEmployeeOfTheMonth"</pre>
1685
                                        inputVariable="candidates"
1686
                                        outputVariable="vote"
1687
                                        isSkipable="yes">
1688
                     <htd:task name="votingTask">
1689
                       <htd:interface operation="vote"
1690
                                       portType="el:votingPT"/>
1691
                       <htd:peopleAssignments>
1692
                         <htd:potentialOwners>
1693
                            <htd:from>$voters/users/user[i]</htd:from>
1694
                         </htd:potentialOwners>
1695
                       </htd:peopleAssignments>
1696
                       <htd:presentationElements/>
1697
                     </htd:task>
1698
                     <b4p:scheduledActions>
1699
                       <br/>b4p:expiration>
1700
                         <br/><b4p:documentation xml:lang="en-US">
1701
                            This people activity expires when not completed
1702
                            within 2 days after having been activated.
1703
                         </bd></bd></ri>
1704
                         <b4p:for>P2D</b4p:for>
1705
                       </bd></bd>
1706
                     </bd></bd></
1707
                   </bd></bd>
1708
                 </extensionActivity>
1709
1710
                 <assign>
1711
                   <copy>
1712
                     <from>$vote</from>
1713
                     <to>$electionResult/votes[i]</to>
1714
                   </copy>
1715
                 </assign>
1716
1717
               </sequence>
1718
             </scope>
1719
           </forEach>
1720
```

```
1721
           <!-- Might be Constellation 5 - standard WS-BPEL 2.0 invoke -->
1722
           <!--
1723
           <invoke name="determineElectionResult" partnerLink="..." operation="..."</pre>
1724
1725
           -->
1726
1727
           <!-- Constellation 2 -->
1728
           <extensionActivity>
1729
             <b4p:peopleActivity name="firstApproval"</pre>
1730
                                  inputVariable="electionResult"
                                 outputVariable="decision">
1731
1732
               <b4p:localTask reference="tns:approveEmployeeOfTheMonth"/>
1733
             </bd></bd>
1734
           </extensionActivity>
1735
1736
           <!-- Constellation 2 with override specifications -->
1737
           <extensionActivity>
1738
             <b4p:peopleActivity name="secondApproval"</pre>
1739
                                 inputVariable="electionResult"
1740
                                 outputVariable="decision">
1741
               <b4p:localTask reference="tns:approveEmployeeOfTheMonth">
1742
                 <htd:peopleAssignments>
1743
                   <htd:excludedOwners>
1744
                     <htd:from>
1745
                       b4p:getActualOwner("tns:firstApproval")
1746
                     </htd:from>
1747
                   </htd:excludedOwners>
1748
                 </htd:peopleAssignments>
1749
               </bd></bd></r>
1750
             </bd></bd>
1751
           </extensionActivity>
1752
1753
           <!-- Constellation 3 -->
1754
           <extensionActivity>
1755
             <b4p:peopleActivity name="notifyEmployees"</pre>
1756
                                 inputVariable="electionResult">
1757
               <b4p:localNotification reference="ta:employeeBroadcast"/>
1758
               <!-- notification is not defined as part of this document,
1759
                    but within a separate one
1760
1761
             </bd></bd>
1762
           </extensionActivity>
1763
1764
           <!-- Constellation 4 -->
1765
           <extensionActivity>
1766
             <b4p:peopleActivity name="prepareInauguralSpeech"</pre>
                                  inputVariable="electionResult"
1767
1768
                                  outputVariable="speech"
1769
                                 isSkipable="no">
1770
               <b4p:remoteTask partnerLink="author"</pre>
1771
                               operation="prepareSpeech"
1772
                               responseOperation="receiveSpeech">
1773
                 <htd:priority>0</htd:priority> <!-- assign highest priority -->
1774
                 <htd:peopleAssignments>
1775
                     <htd:potentialOwners>
1776
                       <htd:from>$electionResult/winner</htd:from>
1777
                     </htd:potentialOwners>
1778
                 </htd:peopleAssignments>
```

D.2 WSDL Definitions

1786

```
1787
      <?xml version="1.0" encoding="UTF-8"?>
1788
      <!--
1789
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1790
1791
      <wsdl:definitions</pre>
1792
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1793
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1794
         xmlns:tns="http://www.example.com/approval"
1795
         targetNamespace="http://www.example.com/approval">
1796
1797
         <wsdl:documentation>
1798
           Example for BPEL4People 1.1 - PeopleActivity Interface Definition
1799
         </wsdl:documentation>
1800
1801
         <!-- Messages -->
1802
         <wsdl:message name="approvalInput">
1803
           <wsdl:part name="parameters" type="xsd:string" />
1804
         </wsdl:message>
1805
         <wsdl:message name="approvalOutput">
1806
           <wsdl:part name="parameters" type="xsd:string" />
1807
         </wsdl:message>
1808
1809
        <!-- Port Type -->
1810
         <wsdl:portType name="approvalPT">
1811
           <wsdl:operation name="approve">
1812
             <wsdl:input message="tns:approvalInput" />
1813
             <wsdl:output message="tns:approvalOutput" />
1814
           </wsdl:operation>
1815
         </wsdl:portType>
1816
1817
      </wsdl:definitions>
```

```
1819
       <?xml version="1.0" encoding="UTF-8"?>
1820
       <!--
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1821
1822
       -->
1823
      <wsdl:definitions</pre>
1824
         xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1825
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1826
         xmlns:plnk="http://docs.oasis-open.org/wsbpe1/2.0/plnktype"
1827
         xmlns:tns="http://www.example.com/election"
1828
         targetNamespace="http://www.example.com/election">
1829
1830
         <wsdl:documentation>
1831
           Example for BPEL4People 1.1 - PeopleActivity Interface Definition
1832
         </wsdl:documentation>
1833
```

```
1834
         <!-- WS-BPEL 2.0 Partner Link Type -->
1835
         <plnk:partnerLinkType name="electionPLT">
1836
           <plnk:role name="electionService" portType="tns:electionPT" />
1837
         </plnk:partnerLinkType>
1838
1839
         <!-- Messages -->
1840
         <wsdl:message name="electionInput">
1841
          <wsdl:part name="parameters" type="xsd:string" />
1842
         </wsdl:message>
1843
         <wsdl:message name="votingInput">
1844
          <wsdl:part name="parameters" type="xsd:string" />
1845
         </wsdl:message>
1846
1847
         <!-- Port Types -->
1848
         <wsdl:portType name="electionPT">
1849
           <wsdl:operation name="elect">
1850
             <wsdl:input message="tns:electionInput" />
1851
           </wsdl:operation>
1852
         </wsdl:portType>
1853
         <wsdl:portType name="votingPT">
1854
           <wsdl:operation name="vote">
1855
             <wsdl:input message="tns:votingInput" />
1856
           </wsdl:operation>
1857
         </wsdl:portType>
1858
1859
      </wsdl:definitions>
```

E. Acknowledgements

1860

1898

1899

1861 The following individuals have participated in the creation of this specification and are gratefully 1862 acknowledged: 1863 1864 Members of the BPEL4People Technical Committee: 1865 Phillip Allen, Microsoft Corporation 1866 Ashish Agrawal, Adobe Systems 1867 Mike Amend, BEA Systems, Inc. Stefan Baeuerle, SAP AG 1868 1869 Charlton Barreto, Adobe Systems 1870 Justin Brunt, TIBCO Software Inc. 1871 Martin Chapman, Oracle Corporation 1872 Luc Clément, Active Endpoints, Inc. 1873 Manoj Das, Oracle Corporation 1874 Alireza Farhoush, TIBCO Software Inc. 1875 Mark Ford, Active Endpoints, Inc. 1876 Sabine Holz, SAP AG 1877 Dave Ings, IBM 1878 Gershon Janssen, Individual 1879 Diane Jordan, IBM 1880 Anish Karmarkar, Oracle Corporation 1881 Ulrich Keil, SAP AG Oliver Kieselbach, SAP AG 1882 1883 Matthias Kloppmann, IBM 1884 Dieter König, IBM 1885 Marita Kruempelmann, SAP AG 1886 Frank Leymann, IBM 1887 Mark Little, Red Hat Alexander Malek, Microsoft Corporation 1888 1889 Ashok Malhotra, Oracle Corporation 1890 Mike Marin, IBM Vinkesh Mehta, Deloitte Consulting LLP 1891 Jeff Mischkinsky, Oracle Corporation 1892 1893 Ralf Mueller, Oracle Corporation Krasimir Nedkov, SAP AG 1894 1895 Benjamin Notheis, SAP AG 1896 Michael Pellegrini, Active Endpoints, Inc. 1897 Hannah Petereit, SAP AG

Karsten Ploesser, SAP AG

Gerhard Pfau, IBM

1901	Alan Rickayzen, SAP AG
1902	Michael Rowley, BEA Systems, Inc.
1903	Ron Ten-Hove, Sun Microsystems
1904	Ivana Trickovic, SAP AG
1905	Alessandro Triglia, OSS Nokalva
1906	Claus von Riegen, SAP AG
1907	Peter Walker, Sun Microsystems
1908	Franz Weber, SAP AG
1909	Prasad Yendluri, Software AG, Inc.
1910	
1911	BPEL4People 1.0 Specification Contributors:
1912	Ashish Agrawal, Adobe
1913	Mike Amend, BEA
1914	Manoj Das, Oracle
1915	Mark Ford, Active Endpoints
1916	Chris Keller, Active Endpoints
1917	Matthias Kloppmann, IBM
1918	Dieter König, IBM
1919	Frank Leymann, IBM
1920	Ralf Müller, Oracle
1921	Gerhard Pfau, IBM
1922	Karsten Plösser, SAP
1923	Ravi Rangaswamy, Oracle
1924	Alan Rickayzen, SAP
1925	Michael Rowley, BEA
1926	Patrick Schmidt, SAP
1927	Ivana Trickovic, SAP
1928	Alex Yiu, Oracle
1929	Matthias Zeller, Adobe
1930	
1931 1932 1933 1934	In addition, the following individuals have provided valuable input into the design of this specification: Dave Ings, Diane Jordan, Mohan Kamath, Ulrich Keil, Matthias Kruse, Kurt Lind, Jeff Mischkinsky, Bhagat Nainani, Michael Pellegrini, Lars Rueter, Frank Ryan, David Shaffer, Will Stallard, Cyrille Waguet, Franz Weber, and Eric Wittmann.

1900

Ravi Rangaswamy, Oracle Corporation

F. Revision History

Revision	Date	Editor	Changes Made
WD-01	2008-03-12	Dieter König	First working draft created from submitted specification
WD-02	2008-03-13	Dieter König	Added specification editors Moved WSDL and XSD into separate artifacts
WD-02	2008-06-25	Ivana Trickovic	Resolution of Issue #8 incorporated into the document/section 5
WD-02	2008-06-28	Dieter König	Resolution of Issue #13 applied to complete document and all separate XML artifacts
WD-02	2008-06-28	Dieter König	Resolution of Issue #21 applied to section 2
			Resolution of Issue #22 applied to sections 2.4.1 and 3.1.1
WD-02	2008-07-06	Vinkesh Mehta	Resolution for Issue #3 applied to sections 2.4.1 (~line 353)
WD-02	2008-07-25	Krasimir Nedkov	Resolution for Issue #18 applied to sections 4.6.2 and 5; Typos correction.
WD-02	2008-07-29	Ralf Mueller	Resolution for Issue #11 applied to section 3.1.2
WD-02	2008-07-29	Luc Clément	Resolution for Issue #10 applied to first paragraph of section 3.3
CD-01-rev-1	2008-10-02	Ralf Mueller	Resolution for Issue #17 and #24 applied to section 2 and 5
CD-01-rev-2	2008-10-07	Michael Rowley	Resolution for Issue #2 applied in section 4.7, and for issue #19 in sections 4.3.1 and 4.4.1.
CD-01-rev-3	2008-10-20	Dieter König	Resolution for Issue #23 applied to section 3.2.1
			Resolution of Issue #6 applied to section 5
CD-01-rev-3	2008-10-20	Vinkesh Mehta	Resolution of issue-12, section 3.2.2, 4.2 font changed to italics for htd:genericHumanRole. Also modified XML artifacts for boel4people.xsd, humantask.xsd, humantask-context.xsd
CD-01-rev-3	2008-12-03	Ralf Mueller	Resolution for Issue #16 applied to sections 1 – 6

Revision	Date	Editor	Changes Made
CD-01-rev-3	2008-12-12	Ravi Rangaswamy	Resolution for Issue #16 applied to sections 7 and appendix B
CD-01-rev-3	2008-12-18	Ravi Rangaswamy	Resolution for Issue #16: Undid changes to appendix B
CD-01-rev-4	2008-12-19	Ralf Mueller	Incorporated review comments from Ivana and Luc for Issue BP-16
CD-02	2009-01-18	Luc Clément	Committee Draft 2
CD-02-rev-1	2009-02-20	Dieter König	Issue 47: added getState() in section 5 Issue 48: abstract BPEL ns in 7.1.1 Issue 50, sections 3 and 5 (htd:→htt:)
CD-02-rev-2	2009-03-11	Ralf Mueller	Issue 76: Changes for RFC2119
CD-03	2009-04-15	Luc Clément	Committee Draft 3
CD-03-rev-2	2009-04-29	Luc Clément	Issue 72: add WS-HumanTask and WS- HumanTask Processor definitions to section 2.3
CD-03-rev3	2009-06-01	Luc Clément	Issue 65
CD-03-rev4	2009-06-02	Michael Rowley	Issue 38, 39
CD-04-rev0	2009-06-17	Luc Clément	Committee Draft 4
CD-04-rev1	2009-06-17	Luc Clément	Acknowledgement update
CD-04-rev2	2009-06-26	Dieter König	Formatting
CD-05-rev0	2009-07-15	Luc Clément	Committee Draft 5
CD-05-rev1	2009-08-08	Luc Clément	Editors update
CD-05-rev2	2009-09-28	Dieter König	Issue 125
CD-05-rev3	2009-10-22	Dieter König	Issue 129 XML artifacts copied back to appendix
CD-05-rev4	2009-11-01	Luc Clément	Issue 131 OASIS Spec QA Checklist updates
CD-06-rev0	2009-11-01	Luc Clément	Committee Draft 6
CD-07	2010-03-03	Luc Clément	Copyright date updates, creation of CD07, and cover page annotation as Public Review 02
CD-08	2010-04-14	Luc Clément	CD08
CD-09	2010-04-26	Luc Clément	CD09 / PRD 03
PRD-03	2010-05-12	Luc Clément	PRD-03 Approved for Public Review