OASIS 🕅

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

Committee Draft 06 / Public Review Draft 01

04 November 2009

Specification URIs:

This Version:

http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cd-06.html http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cd-06.doc (Authoritative format) http://docs.oasis-open.org/bpel4people/bpel4people-1.1-spec-cd-06.pdf

Previous Version:

N/A

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 firstclass 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.oasisopen.org/committees/bpel4people/.

Notices

Copyright © OASIS® 2009. 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 names "OASIS", are trademarks 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	
	2.1 Dependencies on Other Specifications	9
	2.1.1 Namespaces Referenced	
	2.2 Language Extensibility	9
	2.3 Overall Language Structure	
	2.3.1 Syntax	
	2.4 Default use of XPath 1.0 as an Expression Language	12
3	Concepts	13
	3.1 Generic Human Roles	
	3.1.1 Syntax	
	3.1.2 Initialization Behavior	
	3.2 Assigning People	
	3.2.1 Using Logical People Groups	
	3.2.2 Computed Assignment	
	3.3 Ad-hoc Attachments	
4	People Activity	
	4.1 Overall Syntax	
	4.1.1 Properties	
	4.2 Standard Overriding Elements	
	4.3 People Activities Using Local Human Tasks	
	4.3.1 Syntax	
	4.3.2 Examples	
	4.4 People Activities Using Local Notifications	
	4.4.1 Syntax	
	4.4.2 Examples	
	4.5 People Activities Using Remote Human Tasks	
	4.5.1 Syntax	
	4.5.2 Example	
	4.5.3 Passing Endpoint References for Callbacks	
	4.6 People Activities Using Remote Notifications	
	4.6.1 Syntax	
	4.6.2 Example	
	4.7 Elements for Scheduled Actions	
	4.8 People Activity Behavior and State Transitions	
	4.9 Task Instance Data	
	4.9.1 Presentation Data	
	4.9.2 Context Data	
	4.9.3 Operational Data	29

XPath Extension Functions	30
Coordinating Standalone Human Tasks	33
5.1 Protocol Messages from the People Activity's Perspective	33
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	35
2.2 Abstract Process Profile for Observable Behavior	35
'.3 Abstract Process Profile for Templates	36
Conformance	37
Standard Faults	38
Portability and Interoperability Considerations	. 39
BPEL4People Schema	40
Sample	46
0.1 BPEL Definition	47
0.2 WSDL Definitions	52
Acknowledgements	54
Non-Normative Text	. 56
Revision History	57
	XPath Extension Functions Coordinating Standalone Human Tasks

1 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

user interface, and taking into account human behavior. This specification introduces a set of elements
 which extend the standard BPEL elements and enable the modeling of human interactions, which may

9 which extend the standard BPEL elements and enable the modeling of numan interactions, which has 10 range from simple approvals to complex scenarios such as separation of duties, and interactions

- 11 involving ad-hoc data.
- 12 The specification introduces the people activity as a new type of basic activity which enables the
- 13 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
- 16 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.

- 18 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 20 another vendor's environment.
- 21 Interoperability The capability for multiple components (process infrastructure, task infrastructures and
- task list clients) to interact using well-defined messages and protocols. This enables combining components from different vendors allowing seamless execution.
- 24 Out of scope of this specification is how processes with human interactions are deployed or monitored.
- 25 Usually people assignment is accomplished by performing queries on a people directory which has a
- 26 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.

28 **1.1 Terminology**

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

32 **1.2 Normative References**

33 [BPEL4WS 1.1]

- 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-
- 36 128.ibm.com/developerworks/library/specification/ws-bpel/, http://ifr.sap.com/bpel4ws/

37 [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.isi.edu/in-notes/rfc3066.txt

43 [WS-Addr-Core]

44 Web Services Addressing 1.0 - Core, W3C Recommendation, May 2006, available via 45 http://www.w3.org/TR/ws-addr-core

46	[WS-Addr-SOAP]		
47 48	Web Services Addressing 1.0 – SOAP Binding, W3C Recommendation, May 2006, available via http://www.w3.org/TR/ws-addr-soap		
49	[WS-Addr-WSDL]		
50 51	Web Services Addressing 1.0 – WSDL Binding, W3C Working Draft, February 2006, available via http://www.w3.org/TR/ws-addr-wsdl		
52	[WS-BPEL 2.0]		
53 54	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		
55	[WSDL 1.1]		
56 57	Web Services Description Language (WSDL) Version 1.1, W3C Note, available via http://www.w3.org/TR/2001/NOTE-wsdl-20010315		
58	[WS-HumanTask]		
59 60 61	OASIS Committee Draft, "Web Services – Human Task (WS-HumanTask) Specification Version 1.1, CD-06", 04 November 2009, http://docs.oasis-open.org/bpel4people/ws-humantask-1.1-spec-cd-06.doc		
62	[XML Infoset]		
63 64	XML Information Set, W3C Recommendation, available via http://www.w3.org/TR/2001/REC-xml- infoset-20011024/		
65	[XML Namespaces]		
66 67	Namespaces in XML 1.0 (Second Edition), W3C Recommendation, available via http://www.w3.org/TR/REC-xml-names/		
68	[XML Schema Part 1]		
69 70	XML Schema Part 1: Structures, W3C Recommendation, October 2004, available via http://www.w3.org/TR/xmlschema-1/		
71	[XML Schema Part 2]		
72 73	XML Schema Part 2: Datatypes, W3C Recommendation, October 2004, available via http://www.w3.org/TR/xmlschema-2/		
74	[XMLSpec]		
75 76	XML Specification, W3C Recommendation, February 1998, available via http://www.w3.org/TR/1998/REC-xml-19980210		
77	[XPATH 1.0]		
78 79	XML Path Language (XPath) Version 1.0, W3C Recommendation, November 1999, available via http://www.w3.org/TR/1999/REC-xpath-19991116		
80	1.3 Non-Normative References		
81	There are no non-normative references made by this specification.		
82	1.4 Conformance Targets		

83 As part of this specification, the following conformance targets are specified

- 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	٠	WS-HumanTask Definition
92 93		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.
94	٠	WS-HumanTask Processor
95		A WS-HumanTask Processor is any implementation that accepts a WS-HumanTask
96		definition and executes the semantics as defined by the WS-HumanTask 1.1 specification.

97 2 Language Design

The BPEL4People extension is defined in a way that it is layered on top of BPEL so that its features can
 be composed with BPEL features whenever needed. All elements and attributes introduced in this

- 100 extension are made available to both BPEL executable processes and abstract processes.
- 101 This extension introduces a set of elements and attributes to cover different complex human interaction 102 patterns, such as separation of duties, which are not defined as first-class elements.
- 103 Throughout this specification, WSDL and schema elements may be used for illustrative or convenience
- 104 purposes. However, in a situation where those elements or other text within this document contradict the
- separate BPEL4People, WS-HumanTask, WSDL or schema files, it is those files that have precedence
- 106 and not this document.

107 2.1 Dependencies on Other Specifications

- 108 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.

116 2.1.1 Namespaces Referenced

- 117 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
- 121 abstract http://docs.oasis-open.org/wsbpel/2.0/process/abstract
- wsdl http://schemas.xmlsoap.org/wsdl/
- 123 xsd http://www.w3.org/2001/XMLSchema
- xsi http://www.w3.org/2001/XMLSchema-instance

125 **2.2 Language Extensibility**

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

134 **2.3 Overall Language Structure**

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

137 **2.3.1 Syntax**

138 Informal syntax of a BPEL process and scope containing logical people groups, inline human tasks, and139 people activity follows.

```
140
      <bpel:process b4p:shareComments="xsd:boolean"? ...</pre>
141
142
        xmlns:b4p="http://docs.oasis-open.org/ns/bpel4people/bpel4people/200803"
143
        xmlns:htd="http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803">
144
        . . .
145
        <bpel:extensions>
146
          <bpel:extension</pre>
147
            namespace="http://docs.oasis-
148
      open.org/ns/bpel4people/bpel4people/200803"
149
            mustUnderstand="yes"/>
150
          <bpel:extension</pre>
151
            namespace="http://docs.oasis-open.org/ns/bpel4people/ws-
152
      humantask/200803"
153
            mustUnderstand="yes"/>
154
        </bpel:extensions>
155
156
        <bpel:import</pre>
157
          importType="http://docs.oasis-open.org/ns/bpel4people/ws-
158
      humantask/200803" .../>
159
160
161
        <b4p:humanInteractions>?
162
163
          <htd:logicalPeopleGroups/>?
164
            <htd:logicalPeopleGroup name="NCName" reference="OName"?>+
165
166
            </htd:logicalPeopleGroup>
167
          </htd:logicalPeopleGroups>
168
169
          <htd:tasks>?
170
            <htd:task name="NCName">+
171
               . . .
172
            </htd:task>
173
          </htd:tasks>
174
175
          <htd:notifications>?
176
            <htd:notification name="NCName">+
177
               . . .
178
            </htd:notification>
179
          </htd:notifications>
180
181
        </b4p:humanInteractions>
182
183
        <b4p:peopleAssignments>?
184
185
        </b4p:peopleAssignments>
186
187
        . . .
188
        <bpel:extensionActivity>
189
          <b4p:peopleActivity name="NCName" ...>
190
            . . .
191
          </b4p:peopleActivity>
192
        </bpel:extensionActivity>
193
        . . .
```

194 </bpel:process>

- 195 A BPEL4People Definition MUST use BPEL4People extension elements and elements from WS-
- HumanTask namespace. Therefore elements from namespaces BPEL4People and WS-HumanTask
 MUST be understood.
- 198 The element <b4p:humanInteractions> is optional and contains declarations of elements from WS-
- 199 HumanTask namespace, that is <htd:logicalPeopleGroups>, <htd:tasks> and
- 200 <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
- 203 MUST be unique among the names of all logical people groups defined within the
- 204 <b4p:humanInteractions> element.
- 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 <htd:tasks> element.
- 209 The <htd: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
- specifies the name of the notification. The name MUST be unique among the names of all notifications
- 212 defined within the <htd:notifications> element.
- 213 The element <b4p:peopleAssignments> is used to assign people to process-related generic human
- roles. This element is optional. The syntax and semantics are introduced in section 3.1 "Generic Human Roles".
- 216 New activity type <b4p:peopleActivity> is used to model human interactions within BPEL
- 217 processes. The new activity is included in the BPEL activity

 bpel:extensionActivity> which is
- used as wrapper. The syntax and semantics of the people activity are introduced in section 4 "People
 Activity".
- 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
- to any other task within the same scope that is started after the first task completes. When comments
- 223 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 child or descendent scopes. When a scope specifies @b4p:shareComments="true" then child and
- descendent scopes MUST NOT specify @b4p:shareComments="false". However, an individual
- 227 people activity can prevent its tasks' comments from being propagated by specifying
- 228 @dontShareComments="true".
- 229

230 231 <bpel:scope b4p:shareComments="xsd:boolean"? ...> 232 233 <b4p:humanInteractions>? 234 . . . 235 </b4p:humanInteractions> 236 . . . 237 <bpel:extensionActivity> 238 <b4p:peopleActivity name="NCName" dontShareComments="xsd:boolean" ...> 239 240 </b4p:peopleActivity> 241 </bpel:extensionActivity> 242 . . . 243 </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:

```
249 <b4p:documentation xml:lang="xsd:language">
250 ...
251 </b4p:documentation>
```

252 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.

268 3 Concepts

269 Many of the concepts in BPEL4People are inherited from the WS-HumanTask specification so familiarity 270 with this specification is assumed.

271 **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

277

- 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.

290 Business administrators are people allowed to perform administrative actions on the business process,

291 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

294 Breckreopie Demittion MAT demine assignment for this generic numar role. A compliant Breck 295 Processor MUST ensure that at runtime at least one person is associated with this role.

296 **3.1.1 Syntax**

299

300

301

302

297 <b4p:peopleAssignments>?
298

```
<htd:genericHumanRole>+
<htd:from>...</htd:from>
</htd:genericHumanRole>
```

303 <b4p:peopleAssignments>

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

```
306
      <b4p:peopleAssignments>?
307
308
        <b4p:processInitiator>?
309
          <htd:from ...>...</htd:from>
310
        </b4p:processInitiator>
311
312
        <b4p:processStakeholders>?
313
          <htd:from ...>...</htd:from>
314
        </b4p:processStakeholders>
315
```

316 <b4p:businessAdministrators>? 317 <htd:from ...>...</htd:from> 318 </b4p:businessAdministrators> 319 320 </b4p:peopleAssignments>

Only process-related human roles MUST be used within the <b4p:peopleAssignments> element. 321 322 People are assigned to these roles as described in section 3.2 ("Assigning People").

3.1.2 Initialization Behavior 323

324 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 325 326 roles after the end of the initial start activity of the process and before processing other activities or links 327 leaving the start activity. If that initialization fails then the fault b4p:initializationFailure MUST be 328 thrown by a BPEL4People Processor.

3.2 Assigning People 329

To determine who is responsible for acting on a process, a human task or a notification in a certain 330 generic human role, people need to be assigned. People assignment can be achieved in different ways: 331

- 332 Via logical people groups (see 3.2.1 "Using Logical People Groups") •
- Via literals (as introduced section 3.2.2 in [WS-HumanTask]) 333 •
- Via expressions (see 3.2.2 "Computed Assignment") 334 •

335 When specifying people assignments then the data type htt:tOrganizationalEntity defined in 336 [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"). 337

3.2.1 Using Logical People Groups 338

339 This section focuses on describing aspects of logical people groups that are specific to business 340 processes. Logical people groups define which person or set of people can interact with a human task or 341 a notification of a people activity. Details about how logical people groups are used with human tasks and 342 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 343 344 either at the process level or on enclosed scopes. Definitions on inner scopes override definitions on 345 outer scopes or the process respectively.

346 Logical people group definitions can be referenced by multiple people activities. Each logical people 347 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 348

arguments. Whenever a logical people group is referenced for the first time with a given set of unique 349

350 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 351

352 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.

- 353
- 354 In particular, for a logical people group with no parameters, there is a single instance, which MUST be 355 evaluated / resolved when the logical people group is first referenced, and which MAY be re-evaluated / 356 re-resolved when referenced again.
- 357 Hence, using the same logical people group does not necessarily mean that the result of a people guery
- 358 is re-used, but that the same guery is used to obtain a result. If the result of a previous people guery
- 359 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. 360
- 361
- 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.

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

369	<pre><bpel:from b4p:logicalpeoplegroup="NCName"></bpel:from></pre>
370	<b4p:argument ?="" expressionlanguage="anyURI" name="NCName">*</b4p:argument>
371	value
372	
373	
374	
375	<to b4p:logicalpeoplegroup="NCName"></to>

In this form of from-spec and to-spec the b4p:logicalPeopleGroup attribute provides the name of a logical people group. The from-spec variant MAY include zero or more <b4p:argument> elements in order to pass values used in the people query. The expressionLanguage 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.

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

385	To copy to a variable
386	<pre><bpel:to ?="" part="NCName" variable="BPELVariableName"></bpel:to></pre>
387	<pre><bpel:query ?="" querylanguage="anyURI">?</bpel:query></pre>
388	queryContent
389	
390	
391	
392	 To copy to non-message variables and parts of message variables
393	<pre><bpel:to ?="" expressionlanguage="anyURI">expression</bpel:to></pre>
394	
395	To copy to a property
396	<pre><bpel:to property="QName" variable="BPELVariableName"></bpel:to></pre>
397	-
398	To copy to a logical people group
399	<pre><bpel:to b4p:logicalpeoplegroup="NCName"></bpel:to></pre>
400	
401	Using a logical people group in the to-spec of a <bpel:copy> assignment enables a set of people to be</bpel:copy>
402	explicitly assigned. Whenever the logical people group is used after the assignment this assigned set of
403	people is returned. Assigning values to a logical people group overrides what has been defined during
404	deployment. This is true irrespective of any parameters specified for the logical people group.
405	The new form of the to-spec can be used with the following from-spec variants:
406	To copy from a variable
407	<pre><bpel:from ?="" part="NCName" variable="BPELVariableName"></bpel:from></pre>
408	<pre><bpel:query ?="" querylanguage="anyURI">?</bpel:query></pre>
409	queryContent
410	
411	
412	
413	To copy from a property
414	<pre><bpel:from property="QName" variable="BPELVariableName"></bpel:from></pre>
	bpel4people-1.1-spec-cd-06 04 November 200

415	
416	 To copy from non-message variables and parts of message variables
417	<pre><bpel:from ?="" expressionlanguage="anyURI">expression</bpel:from></pre>
418	
419	To copy from a literal value
420 421	<pre><bpel:from></bpel:from></pre>
421	<pre><bpel:literal>literal value</bpel:literal> </pre>
423	
424	To copy from a logical people group
425	<pre><bpel:from b4p:logicalpeoplegroup="NCName"></bpel:from></pre>
426	
427	Below are several examples illustrating the usage of logical people groups in copy assignments. The first
428	example shows assigning the results of the evaluation of a logical people group to a process variable.
429	<pre><bpel:assign name="getVoters"></bpel:assign></pre>
430	<pre><bpel:copy></bpel:copy></pre>
431	<pre><bpel:from b4p:logicalpeoplegroup="voters"></bpel:from></pre>
432	<b4p:argument name="region"></b4p:argument>
433	<pre>\$electionRequest/region</pre>
434	
435	
436	<pre><bpel:to variable="voters"></bpel:to></pre>
437	
438	
439	
440	The next example demonstrates assigning a set of people to a logical people group using literal values.
441 442	<pre><bpel:assign> </bpel:assign></pre>
442 443	<pre><bpel:copy> <bpel:from></bpel:from></bpel:copy></pre>
444	<pre><bpe1:110m <bpe1:literal=""></bpe1:110m></pre>
445	<pre><htt:torganizationalentity></htt:torganizationalentity></pre>
446	<htt:user>Alan</htt:user>
447	<htt:user>Dieter</htt:user>
448	<pre><htt:user>Frank</htt:user></pre>
449	<pre><htt:user>Gerhard</htt:user></pre>
450	<pre><htt:user>Ivana</htt:user></pre>
451	<pre><htt:user>Karsten</htt:user></pre>
452	<pre><htt:user>Matthias</htt:user></pre>
453	<pre><htt:user>Patrick</htt:user></pre>
454	
455	
456	/bpel:from>
457	<pre><bpel:to b4p:logicalpeoplegroup="bpel4peopleAuthors"></bpel:to></pre>
458	
459	

461

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

```
464 <bpel:assign>
465 <bpel:copy>
466 <bpel:from b4p:logicalPeopleGroup="bpel4peopleAuthors" />
467 <bpel:to b4p:logicalPeopleGroup="approvers" />
468 </bpel:copy>
469 </bpel:assign>
```

470 3.2.2 Computed Assignment

471 All computed assignment variants described in [WS-HumanTask] (see section 3.2 "Assigning People" for 472 more details) are supported. In addition, the following variant is possible:

```
473 <htd:genericHumanRole>
474 <bpel:from variable="NCName" part="NCName"? >
475 ...
476 </bpel:from>
477 </htd:genericHumanRole>
```

- 478 The from-spec variant

 the pel:from variable> is used to assign people that have been specified
- 479 using variable of the business process. The data type of the variable MUST be of type
- 480 htt:tOrganizationalEntity.
- 481 All other process context can be accessed using expressions of the following style:
- 482

 spel:from expressionLanguage="anyURI"?>expression</bpl:from>
- 483 with XPath extension functions defined in section 5 "XPath Extension Functions". The
- 484 expressionLanguage attribute specifies the language used in the expression. The attribute is optional.
- 485 If not specified, the default language as inherited from the closest enclosing element that specifies the
- 486 attribute is used.

487 **3.3 Ad-hoc Attachments**

488 Processes can have ad-hoc attachments. It is possible to exchange ad-hoc attachments between people
 489 activities of a process by propagating ad-hoc attachments to and from the process level.

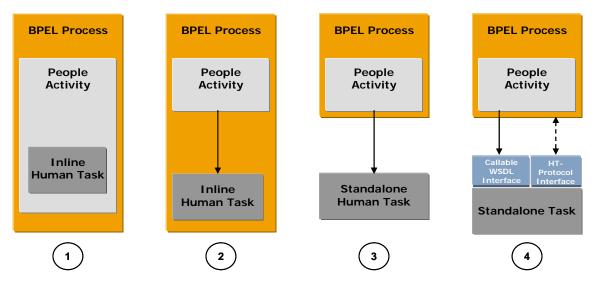
- 490 When a people activity is activated, attachments from earlier tasks and from the process can be
- 491 propagated to its implementing human task. On completion of the human task, its ad-hoc attachments492 can be propagated to the process level, to make them globally available.
- 493 All manipulations of ad-hoc attachments at the process level are instantaneous, and not subject to
- 494 compensation or isolation.

495 **4 People Activity**

496 People activity is a basic activity used to integrate human interactions within BPEL processes. The

497 following figure illustrates different ways in which human interactions (including human tasks and

- 498 notifications) could be integrated.
- 499



500 501

Figure 1: Constellations

502

503 Constellations 1 and 2 show models of interaction in which tasks are defined inline as part of a BPEL 504 process. An *inline task* can be defined as part of a people activity (constellation 1). In this case, the use of 505 the task is limited to the people activity encompassing it. Alternatively, a task can be defined as a top-506 level construct of the BPEL process or scope (constellation 2). In this case, the same task can be used 507 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

509 holds true for notifications.

510 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

512 constellation is similar to constellation 2, except that the definition of the task is done independently of 513 any process. As a result, the task has no direct access to process context. This also holds true for

514 notifications.

515 Constellation 4 shows the use of a standalone task from a different environment. The major difference 516 when compared to constellation 3 is that the task has a Web services callable interface, which is invoked

517 using Web services protocols. In addition, the WS-HumanTask coordination protocol is used to

- 518 communicate between processes and tasks (see section 6 "Coordinating Standalone Human Tasks" for
- 519 more details on the WS-HumanTask coordination protocol). Using this mechanism, state changes are
- 520 propagated between task and process activity, and the process can perform life cycle operations on the
- 521 task, such as terminating it. BPEL4People processes that use tasks in this way are portable across
- 522 different BPEL engines that implement BPEL4People. They are interoperable, assuming that both the
- 523 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
- 525 BPEL4People specifications refer to section 1.1 of WS-HumanTask.

526 4.1 Overall Syntax

```
527
      Definition of people activity:
528
      <bpel:extensionActivity>
529
530
        <b4p:peopleActivity name="NCName" inputVariable="NCName"?
531
          outputVariable="NCName"? isSkipable="xsd:boolean"?
532
          dontShareComments="xsd:boolean"?
533
          standard-attributes>
534
535
          standard-elements
536
           <htd:task>...</htd:task>
537
538
            <b4p:localTask>...</b4p:localTask>
539
            <b4p:remoteTask>...</b4p:remoteTask>
540
            <htd:notification>...</htd:notification>
541
            <b4p:localNotification>...</b4p:localNotification>
542
            <b4p:remoteNotification>...</b4p:remoteNotification>
543
544
545
          <b4p:scheduledActions>? ...</b4p:scheduledActions>
546
547
          <bpel:toParts>?
548
            <bpel:toPart part="NCName" fromVariable="BPELVariableName" />+
549
          </bpel:toParts>
550
551
          <bpel:fromParts>?
552
            <bpel:fromPart part="NCName" toVariable="BPELVariableName" />+
553
          </bpel:fromParts>
554
555
          <b4p:attachmentPropagation fromProcess="all none"
556
            toProcess="all|newOnly|none" />?
557
558
        </b4p:peopleActivity>
559
560
     </bpel:extensionActivity>
```

561 4.1.1 Properties

562 The <b4p:peopleActivity> element is enclosed in the BPEL extensionActivity and has the 563 following attributes and elements:

- inputVariable: This attribute refers to a process variable which is used as input of the WSDL operation of a task or notification. The process variable in the BPEL4People Definition MUST have a WSDL message type. This attribute is optional. If this attribute is not present the

 567
- 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 <bpel:fromParts> element MUST be used. The outputVariable
 attribute MUST NOT be used if the people activity uses a notification.
- 573 isSkipable: This attribute indicates whether the task associated with the activity can be
 574 skipped at runtime or not. This is propagated to the task level. This attribute is optional. The
 575 default for this attribute is "no".
- dontShareComments: This attribute, if set to "true", indicates that comments that are added to 577 the task associated with this people activity MUST NOT be propagated to any other task.

578	•	standard-attributes: The activity makes available all BPEL's standard attributes.
579	•	standard-elements: The activity makes available all BPEL's standard elements.
580 581 582		 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".
583 584 585		 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"
586 587 588		 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".
589 590 591		 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".
592 593 594		 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".
595 596 597	•	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".
598 599	•	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".
600 601 602 603	•	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 <bpel:toparts> element and the inputVariable attribute are mutually exclusive.</bpel:toparts>
604 605 606 607 608	•	<pre>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.</bpel:fromparts></pre>
609 610 611 612 613 614 615 616 617	•	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.

618 **4.2 Standard Overriding Elements**

619 Certain properties of human tasks and notifications can be specified on the process level as well as on 620 local and remote task definitions and notification definitions allowing the process to override the original 621 human task and notification definitions respectively. This increases the potential for reuse of tasks and 622 notifications. Overriding takes place upon invocation of the Web service implemented by the human task 623 (or notification) via the advanced interaction protocol implemented by both the process and the task (or 624 notification).

625 The following elements can be overridden:

people assignments

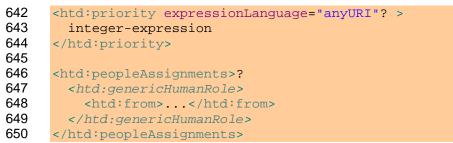
626

627 • priority

628 People assignments can be specified on remote and local human tasks and notifications. As a

629 consequence, the invoked task receives the results of people queries performed by the business process

- 630 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
- 633 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
- 635 generic human roles defined at the task level provide the default. The same applies to people
- 636 assignments on remote and local notifications.
- 637 The task's originator is set to the process stakeholder.
- 638 Priority of tasks and notifications can be specified on remote and local human tasks and notifications. If 639 specified, it overrides the original priority of the human task (or notification).
- 640 *Standard-overriding-elements* is used in the syntax below as a shortened form of the following list of 641 elements:



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

661 **4.3.1 Syntax**

```
662
      <b4p:peopleActivity inputVariable="NCName"? outputVariable="NCName"?</pre>
663
        isSkipable="xsd:boolean"? standard-attributes>
664
        standard-elements
665
666
        ( <htd:task>...</htd:task>
667
          <b4p:localTask reference="QName">
668
            standard-overriding-elements
669
          </b4p:localTask>
670
        )
671
672
      </b4p:peopleActivity>
673
```

674 **Properties**

675 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

679 execution of the enclosed elements, but only provide more current values when the enclosed elements

- 680 choose to re-evaluate the expressions.
- 681 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
- 683 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".

685 4.3.2 Examples

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

687	<b4p:peopleactivity <="" inputvariable="candidates" th=""></b4p:peopleactivity>
688	outputVariable="vote"
689	isSkipable="yes">
690	<htd:task></htd:task>
691	<htd:peopleassignments></htd:peopleassignments>
692	<htd:potentialowners></htd:potentialowners>
693	<htd:from>\$voters/users/user[i]</htd:from>
694	
695	
696	
697	<b4p:scheduledactions></b4p:scheduledactions>
698	<b4p:expiration></b4p:expiration>
699	<b4p:documentation xml:lang="en-US"></b4p:documentation>
700	This people activity expires when not completed
701	within 2 days after having been activated.
702	
703	<b4p:for>P2D</b4p:for>
704	
705	
706	

- 707
- The following code shows a people activity referring to an inline task defined in the BPEL4Peopleprocess.

```
710 <extensionActivity>
711 <br/>
712 <br/>
713 <br/>
714 </br/>
714 </br/>
715 </extensionActivity>
715 </extensionActivity>
```

716 **4.4 People Activities Using Local Notifications**

- 717 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.

726 **4.4.1 Syntax**

```
727
     <b4p:peopleActivity name="NCName"? inputVariable="NCName"?</pre>
728
        standard-attributes>
729
        standard-elements
730
731
        ( <htd:notification>...</htd:notification>
732
         <b4p:localNotification reference="QName">
733
            standard-overriding-elements
734
          </b4p:localNotification>
735
        )
736
      </b4p:peopleActivity>
```

737

738 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 <b4p:localNotification> is used to refer to a notification enclosed in the BPEL4People

746 Definition (a BPEL scope or the process scope) or a standalone notification provided by the same

environment. Attribute reference provides the QName of the notification. The attribute is mandatory.

748 The element MAY contain standard overriding elements explained in section 4.2 "Standard Overriding

749 Elements".

750 **4.4.2 Examples**

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

```
752
     <bpel:extensionActivity>
753
        <b4p:peopleActivity name="notifyEmployees"
754
                            inputVariable="electionResult">
          <htd:localNotification reference="task:employeeBroadcast"/>
755
          <!-- notification is not defined as part of this document,
756
757
              but within a separate one
758
          -->
759
        </b4p:peopleActivity>
760
     </bpel:extensionActivity>
```

761 **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.

773 **4.5.1 Syntax**

774	<b4p:remotetask< th=""></b4p:remotetask<>
775	partnerLink="NCName"
776	operation="NCName"
777	responseOperation="NCName"?>
778	
779	standard-overriding-elements
780	
781	
782	~

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.

789 4.5.2 Example

790	<pre><bpel:extensionactivity></bpel:extensionactivity></pre>
791	<b4p:peopleactivity <="" name="prepareInauguralSpeech" th=""></b4p:peopleactivity>
792	inputVariable="electionResult"
793	outputVariable="speech"
794	isSkipable="no">
795	<b4p:remotetask <="" partnerlink="author" th=""></b4p:remotetask>
796	operation="prepareSpeech"
797	responseOperation="receiveSpeech">
798	<htd:priority>0</htd:priority> assign highest priority
799	<htd:peopleassignments></htd:peopleassignments>
800	<htd:potentialowners></htd:potentialowners>
801	<htd:from>\$electionResult/winner</htd:from>
802	
803	
804	
805	
806	

807 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

819 message is in fact the response for an outstanding request (see [WS-HumanTask] section 8.2 for the

820 definition of the metadata). Finally, such an endpoint reference MUST specify identifying data to allow the 821 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
- 827 with an appropriate endpoint reference.
- 828 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.
- 832 In both cases (a request-response <b4p:remoteTask> as well as a <b4p:remoteTask> using two
- 833 one-ways) the <b4p:remoteTask> activity is blocking. That is, the normal processing of a
- 834 <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
- 836 Processor will signal that to the BPEL4People Processor and an b4p:nonRecoverableError fault
- 837 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
- 843 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.

847 **4.6.1 Syntax**

848	<b4p:remotenotification< th=""></b4p:remotenotification<>
849	partnerLink="NCName"
850	operation="NCName">
851	
852	standard-overriding-elements
853	
854	

855 4.6.2 Example

856

<b

857	<b4p:peopleactivity <="" name="notifyEmployees" th=""></b4p:peopleactivity>
858	inputVariable="electionResult">
859	<b4p:remotenotification <="" partnerlink="employeeNotification" th=""></b4p:remotenotification>
860	operation="receiveElectionResult">
861	<htd:priority>5</htd:priority> assign moderate priority
862	<htd:peopleassignments></htd:peopleassignments>
863	<htd:recipients></htd:recipients>
864	<htd:from>\$voters</htd:from>
865	
866	
867	
868	
869	

870 4.7 Elements for Scheduled Actions

871 Scheduled actions allow the specification of determining when a task needs to change its state. The 872 following scheduled actions are defined:

873 **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.

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 878 879 either the period of time after which the task expires or the point in time when the task expires. The time 880 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 881 MUST change the task state to Exited. Additional user-defined actions MUST NOT be performed. The 882 883 default value is infinity, i.e. the task never expires. If the expiration time is defined as a point in time and 884 the task is created after that point in time the BPEL4People Processor MUST change the task state to 885 Exited. Note that deferred activation does not impact expiration. Therefore the task MAY expire even 886 before being activated.

887 Element <b4p:scheduledActions> is used to include the definition of all scheduled actions within the 888 task definition. If present, at least one scheduled activity MUST be defined in the BPEL4People Definition.

889 **Syntax:**

890	<b4p:scheduledactions>?</b4p:scheduledactions>
891	
892	<b4p:deferactivation>?</b4p:deferactivation>
893	<pre>(<b4p:for ?="" expressionlanguage="anyURI"></b4p:for></pre>
894	duration-expression
895	
896	<b4p:until ?="" expressionlanguage="anyURI"></b4p:until>
897	deadline-expression
898	
899)
900	
901	
902	<b4p:expiration>?</b4p:expiration>
903	<pre>(<b4p:for ?="" expressionlanguage="anyURI"></b4p:for></pre>
904	duration-expression
905	
906	<b4p:until ?="" expressionlanguage="anyURI"></b4p:until>
907	deadline-expression
908	
909)
910	
911	
912	
913	

914 Properties

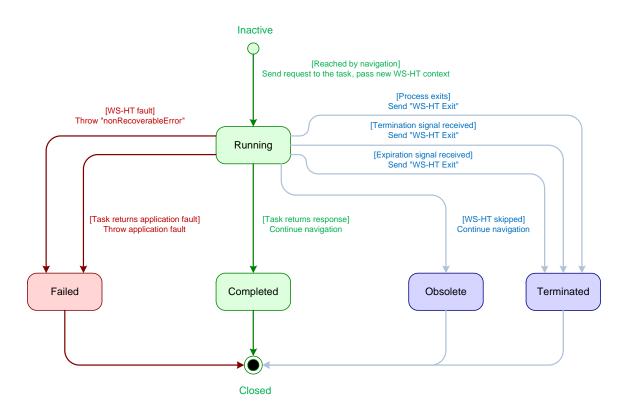
915 The <b4p:scheduledActions> element has the following optional elements:

- 916 b4p:deferActivation: The element is used to specify activation time of the task. It includes
 917 the following elements:
- 918ob4p:for: The element is an expression which specifies the period of time (duration)919after which the task reaches state Ready (in case of explicit claim) or state Reserved (in920case of implicit claim). The absolute time of this transition is computed by adding the921specified duration to the time at which the people activity begins execution.

922 923	 b4p:until: The element is an expression which specifies the point in time when the task reaches state <i>Ready</i> or state <i>Reserved</i>.
924 925	Elements <b4p:for> and <b4p:until> are mutually exclusive. There MUST be at least one <b4p:for> or <b4p:until> element.</b4p:until></b4p:for></b4p:until></b4p:for>
926 927	• b4p:expiration: The element is used to specify the expiration time of the task when the task becomes obsolete:
928 929 930	 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.
931 932	 b4p:until: The element is an expression which specifies the point in time when the task expires.
933 934	Elements <b4p:for> and <b4p:until> are mutually exclusive. There MUST be at least one <b4p:for> or <b4p:until> element.</b4p:until></b4p:for></b4p:until></b4p:for>
935 936 937	The language used in expressions is specified using the expressionLanguage attribute. This attribute is optional. If not specified, the default language as inherited from the closest enclosing element that specifies the attribute is used.
938 939	If specified, the scheduledActions element MUST NOT be empty, that is one of the elements
939	b4p:deferActivation and b4p:expiration MUST be defined.
939 940	Example:
940 941	
940 941 942	Example: <b4p:scheduledactions></b4p:scheduledactions>
940 941 942 943	Example: <b4p:scheduledactions> <b4p:deferactivation></b4p:deferactivation></b4p:scheduledactions>
940 941 942 943 944	Example: <b4p:scheduledactions> <b4p:deferactivation> <b4p:documentation xml:lang="en-US"></b4p:documentation></b4p:deferactivation></b4p:scheduledactions>
940 941 942 943 944 945	Example: <b4p:scheduledactions> <b4p:deferactivation> <b4p:documentation xml:lang="en-US"> Activation of this task is deferred until the time specified</b4p:documentation></b4p:deferactivation></b4p:scheduledactions>
940 941 942 943 944 945 946	<pre>Example:</pre>
940 941 942 943 944 945 946 946 947	<pre>Example: <b4p:scheduledactions></b4p:scheduledactions></pre>
940 941 942 943 944 945 946 947 948	<pre>Example: <b4p:scheduledactions></b4p:scheduledactions></pre>
940 941 942 943 944 945 946 946 947	<pre>Example: <b4p:scheduledactions></b4p:scheduledactions></pre>
940 941 942 943 944 945 946 947 948 949	<pre>Example: <b4p:scheduledactions></b4p:scheduledactions></pre>
940 941 942 943 945 946 947 948 949 950 951 952	<pre>Example: <b4p:scheduledactions> <b4p:deferactivation> <b4p:documentation xml:lang="en-US"> Activation of this task is deferred until the time specified in its input data. </b4p:documentation> <b4p:until>htd:getInput()/activateAt</b4p:until> </b4p:deferactivation> <b4p:expiration> <b4p:documentation xml:lang="en-US"></b4p:documentation></b4p:expiration></b4p:scheduledactions></pre>
940 941 942 943 944 945 946 947 948 949 950 951 952 953	<pre>Example: <bdp:scheduledactions> <bdp:deferactivation> <bdp:documentation xml:lang="en-US"> Activation of this task is deferred until the time specified in its input data. </bdp:documentation> <bdp:until>htd:getInput()/activateAt</bdp:until> </bdp:deferactivation> <bdp:expiration> <bdp:deferactivation xml:lang="en-US"> This task expires when not completed within 14 days after</bdp:deferactivation></bdp:expiration></bdp:scheduledactions></pre>
940 941 942 943 944 945 946 947 948 947 948 949 950 951 952 953 954	<pre>Example: <bdp:scheduledactions> <bdp:deferactivation> <bdp:documentation xml:lang="en-US"> Activation of this task is deferred until the time specified in its input data. </bdp:documentation> <bdp:until>htd:getInput()/activateAt</bdp:until> </bdp:deferactivation> <bdp:expiration> <bdp:deferactivation xml:lang="en-US"> This task expires when not completed within 14 days after having been activated.</bdp:deferactivation></bdp:expiration></bdp:scheduledactions></pre>
940 941 942 943 944 945 946 947 948 947 948 949 950 951 952 953 954 955	<pre>Example: <bdp:deferactivation> <bdp:deferactivation> <bdp:documentation xml:lang="en-US"> Activation of this task is deferred until the time specified in its input data. </bdp:documentation> <bdp:until>htd:getInput()/activateAt</bdp:until> </bdp:deferactivation> <bdp:expiration> <bdp:documentation xml:lang="en-US"> This task expires when not completed within 14 days after having been activated. </bdp:documentation></bdp:expiration></bdp:deferactivation></pre>
940 941 942 943 944 945 946 947 948 947 948 949 950 951 952 953 954 955 956	<pre>Example: <bdp:deferactivation> <bdp:deferactivation> <bdp:documentation xml:lang="en-US"> Activation of this task is deferred until the time specified in its input data. </bdp:documentation> <bdp:until>htd:getInput()/activateAt</bdp:until> </bdp:deferactivation> <bdp:expiration> <bdp:documentation xml:lang="en-US"> This task expires when not completed within 14 days after having been activated. </bdp:documentation> <bdp:documentation> <bdp:documentation> <bdp:documentation> <bdp:documentation> <bdp:documentation> <bdp:documentation> <bdp:documentation> <bdp:for>P14D</bdp:for></bdp:documentation></bdp:documentation></bdp:documentation></bdp:documentation></bdp:documentation></bdp:documentation></bdp:documentation></bdp:expiration></bdp:deferactivation></pre>
940 941 942 943 944 945 946 947 948 947 948 949 950 951 952 953 954 955	<pre>Example: <bdp:deferactivation> <bdp:deferactivation> <bdp:documentation xml:lang="en-US"> Activation of this task is deferred until the time specified in its input data. </bdp:documentation> <bdp:until>htd:getInput()/activateAt</bdp:until> </bdp:deferactivation> <bdp:expiration> <bdp:documentation xml:lang="en-US"> This task expires when not completed within 14 days after having been activated. </bdp:documentation></bdp:expiration></bdp:deferactivation></pre>

960 **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.



963

964

Figure 2: State diagram of the people activity

965

966 When the process execution instantiates a people activity this activity triggers the creation of a task in 967 state *Running*. Upon receiving a response from the task, the people activity completes successfully and 968 its state changes into the final state *Completed*.

969 If the task returns a fault, the people activity completes unsuccessfully and moves to final state *Failed* and

970 the fault is thrown in the scope enclosing the people activity. If the task experiences a non-recoverable

- 971 error, the people activity completes unsuccessfully and the standard fault nonRecoverableError is 972 thrown in the enclosing scope.
- 973 The people activity goes to final state *Obsolete* if the task is skipped.
- 974 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.
- 977 If the task expires, the people activity is terminated prematurely and the associated task exits. In this case
- 978 the standard fault b4p:taskExpired is thrown in the enclosing scope. When the process exits the
- 979 people activity will also be terminated and the associated task is exited.

980 4.9 Task Instance Data

981 As defined by [WS-HumanTask], task instance data falls into the categories presentation data, context 982 data, and operational data. Human tasks defined as part of a BPEL4People compliant business process

983 have a superset of the instance data defined in [WS-HumanTask].

984 4.9.1 Presentation Data

985 The presentation data of tasks defined as part of a BPEL4People compliant business process is 986 equivalent to that of a standalone human task.

987 **4.9.2 Context Data**

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

- 990 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

994 **4.9.3 Operational Data**

995 The operational data of tasks that is defined as part of a BPEL4People compliant business process is 996 equivalent to that of a standalone human task.

997 **5 XPath Extension Functions**

This section introduces XPath extension functions that are provided to be used within the definition of a

999 BPEL4People business process to access process context. Definition of these XPath extension functions 1000 is provided in the table below. Input parameters that specify peopleActivity name MUST be literal strings.

1001 This restriction does not apply to other parameters. Because XPath 1.0 functions do not support returning

1002 faults, an empty node set is returned in the event of an error.

1003

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.	Out the process initiator <pre>(htt:tUser)</pre>
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.	<pre>In people activity name (xsd:string) Out the actual owner (htt:tUser)</pre>
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.	<pre>In people activity name (xsd:string) Out the task initiator (user id as htt:user)</pre>
getTaskStakeholders	Returns the stakeholders of the task. It MUST evaluate to an empty htt:organization alEntity in case of an error.	<pre>In people activity name (xsd:string) Out task stakeholders (htt:organizationalEnt ity)</pre>
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.	<pre>In people activity name (xsd:string) Out potential owners (htt:organizationalEnt ity)</pre>
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.	<pre>In people activity name (xsd:string) Out business administrators (htt:organizationalEnt ity)</pre>
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.	In • people activity name (xsd:string) Out • priority (htt:tPriority)

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)

1004

1005 XPath functions accessing data of a human task only guarantee to return data once the corresponding1006 task has reached a final state.

6 Coordinating Standalone Human Tasks 1007

1008 Using the WS-HT coordination protocol introduced by [WS-HumanTask] (see section 7 "Interoperable

1009 Protocol for Advanced Interaction with Human Tasks" for more details) to control the autonomy and life 1010 cycle of human tasks, a BPEL process with a people activity can act as the parent application for remote human tasks.

1011

1012

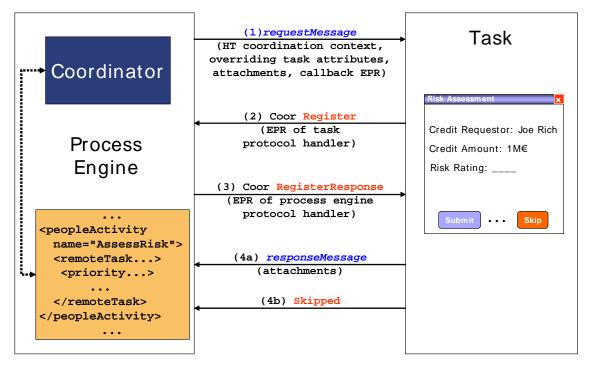


Figure 3: Message exchange between a people activity and a human task

1013

1014 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 1015 activity is the same as for a people activity with an inline human task. That behavior is achieved by 1016 coordinating the remote human task via the WS-HT coordination protocol. 1017

6.1 Protocol Messages from the People Activity's Perspective 1018

1019 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. 1020

1021 1. When the process execution reaches a people activity and determines that this activity can be 1022 executed, the BPEL4People Processor MUST create a WS-HT coordination context associated 1023 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, 1024 people assignments, the skipable indicator and the task's expiration time, are sent. Also the 1025 BPEL4People Processor MAY propagate ad-hoc attachments from the process. All this 1026 information is sent as part of the header fields of the requesting message. These header fields as 1027 well as a corresponding mapping to SOAP headers are discussed in [WS-HumanTask]. 1028

- 1029 2. 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 1030 1031 from the human task. 1032 3. When a response message is received from the task that indicates a fault of the task, the people 1033 activity faults. The fault MUST be thrown in the scope of the people activity. 1034 4. When protocol message fault is received, the fault nonRecoverableError MUST be thrown in the scope enclosing the people activity. 1035 1036 5. When protocol message skipped is received, the people activity MUST move to state Obsolete. 1037 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. 1038 7. When the people activity is terminated, protocol message exit MUST be sent to the task. 1039 1040 8. When the process encounters an <exit> activity, protocol message exit MUST be sent to the task. 1041 1042 The following table summarizes this behavior, the protocol messages sent, and their direction, i.e., 1043 whether a message is sent from the people activity to the task ("out" in the column titled Direction) or vice
- 1044 versa ("in").
- 1045

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>

1046 7 BPEL Abstract Processes

- 1047 BPEL abstract processes are indicated by the namespace "http://docs.oasis-
- 1048 open.org/wsbpel/2.0/process/abstract". All constructs defined in BPEL4People extension
- 1049 namespaces MAY appear in abstract processes.

1050 7.1 Hiding Syntactic Elements

Opaque tokens defined in BPEL (activities, expressions, attributes and from-specs) MAY be used in
 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.

1054 7.1.1 Opaque Activities

- 1055 BPEL4people does not change the way opaque activities can be replaced by an executable activity in an 1056 executable completion of an abstract process, that is, an <abstract:opaqueActivity>MAY also
- 1057 serve as a placeholder for a <bpel:extensionActivity> containing a <b4p:peopleActivity>.

1058 7.1.2 Opaque Expressions

1059 Any expression introduced by BPEL4People MAY be made opaque. In particular, the following 1060 expressions MAY have the opaque="yes" attribute:

1061 <htd:argument name="NCName" expressionLanguage="anyURI"? opaque="yes" />

1062 <htd:priority expressionLanguage="anyURI" opaque="yes" />

1063 <b4p:for expressionLanguage="anyURI"? opaque="yes" />

1064 <b4p:until expressionLanguage="anyURI"? opaque="yes" />

1065 7.1.3 Opaque Attributes

1066 Any attribute introduced by BPEL4People MAY have an opaque value "##opaque" in an abstract 1067 process.

1068 7.1.4 Opaque From-Spec

- 1069 In BPEL, any from-spec in an executable process can be replaced by an opaque from-spec
- $1070 \qquad \texttt{copaqueFrom/s in an abstract process. This already includes any BPEL from-spec extended with the}$
- 1071 BPEL4People b4p:logicalPeopleGroup="NCName" attribute. In addition, the extension from-spec
- 1072 <htd:from> MAY also be replaced by an opaque from-spec in an abstract process.

1073 **7.1.5 Omission**

- 1074 In BPEL, omittable tokens are all attributes, activities, expressions and from-specs which are both (1)
- 1075 syntactically required by the Executable BPEL XML Schema, and (2) have no default value. This rule also
- 1076 applies to BPEL4People extensions in abstract processes. For example, <b4p:localTask
- 1077 reference="##opaque"> is equivalent to <b4p:localTask>.

1078 7.2 Abstract Process Profile for Observable Behavior

- 1079 The Abstract Process Profile for Observable Behavior, indicated by the process attribute
- 1080 abstractProcessProfile="http://docs.oasis-
- 1081 open.org/wsbpel/2.0/process/abstract/ap11/2006/08", provides a means to create precise
- 1082 and predictable descriptions of observable behavior of the service(s) provided by an executable process.

- 1083 The main application of this profile is the definition of business process contracts; that is, the behavior
- 1084 followed by one business partner in the context of Web services exchanges. A valid completion has to
- 1085 follow the same interactions as the abstract process, with the partners that are specified by the abstract
- 1086 process. The executable process can, however, perform additional interaction steps relating to other 1087 partners. Likewise, the executable process can perform additional human interactions. Beyond the
- 1087 partners. Likewise, the executable process can perform additional human interactions. Beyond the 1088 restrictions defined in WS-BPEL 2.0, the use of opacity is not restricted in any way for elements and
- 1089 attributes introduced by BPEL4People.

1090 7.3 Abstract Process Profile for Templates

- 1091 The Abstract Process Profile for Templates, indicated by the process attribute
- 1092 abstractProcessProfile="http://docs.oasis-
- 1093 open.org/wsbpel/2.0/process/abstract/simple-template/2006/08", allows the definition
- 1094 of Abstract Processes which hide almost any arbitrary execution details and have explicit opaque 1095 extension points for adding behavior.
- 1096 This profile does not allow the use of omission shortcuts but the use of opacity is not restricted in any
- 1097 way. For abstract processes belonging to this profile, this rule is extended to the elements and attributes
- 1098 introduced by BPEL4People.

1099 8 Conformance

1100 The XML schema pointed to by the RDDL document at the namespace URI, defined by this specification, 1101 are considered to be authoritative and take precedence over the XML schema defined in the appendix of

- 1102 this document.
- 1103

1104 There are four conformance targets defined as part of this specification: a BPEL4People Definition, a 1105 BPEL4People Processor, a WS-HumanTask Definition and a WS-HumanTask Processor (see section 1106 2.3). In order to claim conformance with BPEL4People 1.1, the conformance targets MUST comply with 1107 all normative statements in the BPEL4People and the WS-HumanTask specification, notably all MUST 1108 statements have to be implemented.

1109 A. Standard Faults

1110 The following list specifies the standard faults defined within the BPEL4People specification. All standard 1111 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

1113 The following section illustrates the portability and interoperability aspects of the various usage

- 1114 constellations of BPEL4People with WS-HumanTask as described in Figure 1:
- 1115

1116 Portability - The ability to take design-time artifacts created in one vendor's environment and use them in 1117 another vendor's environment. Constellations one and two provide portability of BPEL4People processes

1118 with embedded human interactions in. Constellations three and four provide portability of BPEL4People 1119 processes with referenced human interactions.

1120

1121 Interoperability - The capability for multiple components (process engine, task engine and task list client)
 1122 to interact using well-defined messages and protocols. This enables to combine components from
 1123 different vendors allowing seamless execution.

- 1124 Constellation four achieves interoperability between process and tasks from different vendor
- 1125 implementations.
- 1126
- 1127 Constellation 1

1128 Task definitions are defined inline of the people activities. Usage in this manner is typically for self-

1129 contained people activities, whose tasks definitions are not intended to be reused elsewhere in the

1130 process or across multiple processes. This format will also provide scoping of the task definition since it

1131 will not be visible or accessible outside the people activity in which it is contained. Portability for this

- 1132 constellation requires support of both WS-HumanTask and BPEL4People artifacts using the inline task
- 1133 definition format. Since the process and task interactions are combined in one component, interoperability
- 1134 requirements are limited to those between the task list client and the infrastructure.
- 1135
- 1136 Constellation 2

1137 Similar to constellation 1, but tasks are defined at the process level. This allows task definitions to be

1138 referenced from within people activities enabling task reuse. Portability for this constellation requires

1139 support of both WS-HumanTask and BPEL4People artifacts using the process level scoped task

1140 definition format. Since the process and task interactions are combined in one component, interoperability

1141 requirements are limited to those between the task list client and the infrastructure.

- 1142
- 1143 Constellation 3

1144 In this constellation, the task and people activity definitions are defined as separate artifacts and execute 1145 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

1147 task components are implemented by the same vendor, interoperability requirements are limited to those

- 1148 between the task list client and the infrastructure.
- 1149
- 1150 Constellation 4

1151 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

1154 process infrastructures from different vendors is achieved using the WS-HumanTask coordination

1155 protocol.

1156 C. BPEL4People Schema

```
1157
      <?xml version="1.0" encoding="UTF-8"?>
1158
       <!--
1159
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1160
       -->
1161
       <xsd:schema
1162
         targetNamespace="http://docs.oasis-
1163
       open.org/ns/bpel4people/bpel4people/200803"
1164
         xmlns="http://docs.oasis-open.org/ns/bpel4people/bpel4people/200803"
1165
         xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
1166
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1167
         xmlns:htd="http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803"
1168
         elementFormDefault="qualified"
1169
         blockDefault="#all">
1170
1171
         <xsd:annotation>
1172
           <xsd:documentation>
1173
             XML Schema for BPEL4People 1.1 - WS-BPEL 2.0 Extension for Human Task
1174
      Interactions
1175
           </xsd:documentation>
1176
         </xsd:annotation>
1177
1178
         <!-- other namespaces -->
1179
         <xsd:import namespace="http://www.w3.org/XML/1998/namespace"</pre>
1180
           schemaLocation="http://www.w3.org/2001/xml.xsd" />
1181
         <xsd:import namespace="http://docs.oasis-open.org/ns/bpel4people/ws-</pre>
1182
       humantask/200803"
1183
           schemaLocation="ws-humantask.xsd" />
1184
         <xsd:import namespace="http://docs.oasis-</pre>
1185
       open.org/wsbpel/2.0/process/executable"
1186
           schemaLocation="http://docs.oasis-
1187
       open.org/wsbpel/2.0/0S/process/executable/ws-bpel_executable.xsd" />
1188
1189
         <!-- base types for extensible elements -->
1190
         <xsd:complexType name="tExtensibleElements">
1191
           <xsd:sequence>
1192
             <xsd:element name="documentation" type="tDocumentation"</pre>
1193
               minOccurs="0" maxOccurs="unbounded" />
1194
             <xsd:any namespace="##other" processContents="lax" minOccurs="0"</pre>
1195
               maxOccurs="unbounded" />
1196
           </xsd:sequence>
1197
           <xsd:anyAttribute namespace="##other" processContents="lax" />
1198
         </xsd:complexType>
1199
         <xsd:complexType name="tExtensibleMixedNamespaceElements">
1200
           <xsd:sequence>
1201
             <xsd:element name="documentation" type="tDocumentation"</pre>
1202
               minOccurs="0" maxOccurs="unbounded" />
1203
             <xsd:element name="extensions" type="tExtensions" minOccurs="0" />
1204
           </xsd:sequence>
1205
           <xsd:anyAttribute namespace="##other" processContents="lax" />
1206
         </xsd:complexType>
1207
         <xsd:complexType name="tDocumentation" mixed="true">
1208
           <xsd:sequence>
1209
             <xsd:any namespace="##other" processContents="lax" minOccurs="0"</pre>
```

```
1210
               maxOccurs="unbounded" />
1211
           </xsd:sequence>
1212
           <xsd:attribute ref="xml:lang" />
1213
         </xsd:complexType>
1214
         <xsd:complexType name="tExtensions">
1215
           <xsd:sequence>
1216
             <xsd:any namespace="##other" processContents="lax" minOccurs="0"</pre>
1217
               maxOccurs="unbounded" />
1218
           </xsd:sequence>
1219
         </xsd:complexType>
1220
1221
         <!-- element "humanInteractions" to be used within "bpel:process" -->
1222
         <xsd:element name="humanInteractions" type="tHumanInteractions" />
1223
         <xsd:complexType name="tHumanInteractions">
1224
           <xsd:complexContent>
1225
             <xsd:extension base="tExtensibleMixedNamespaceElements">
1226
               <xsd:sequence>
1227
                 <xsd:element ref="htd:logicalPeopleGroups" minOccurs="0" />
1228
                 <xsd:element ref="htd:tasks" minOccurs="0" />
1229
                 <xsd:element ref="htd:notifications" minOccurs="0" />
1230
               </xsd:sequence>
1231
             </xsd:extension>
1232
           </xsd:complexContent>
1233
         </xsd:complexType>
1234
1235
         <!-- element "peopleAssignments" to be used within "bpel:process" -->
1236
         <xsd:element name="peopleAssignments" type="tPeopleAssignments" />
1237
         <xsd:complexType name="tPeopleAssignments">
1238
           <xsd:complexContent>
1239
             <rr><rd:extension base="tExtensibleElements"></r>
1240
               <xsd:sequence>
1241
                 <xsd:element ref="genericHumanRole" minOccurs="1"</pre>
1242
      maxOccurs="unbounded" />
1243
               </xsd:sequence>
1244
             </xsd:extension>
1245
           </xsd:complexContent>
1246
         </xsd:complexType>
1247
1248
         <!-- element "genericHumanRole" within BPEL4People -->
1249
         <xsd:element name="genericHumanRole"</pre>
1250
       type="htd:tGenericHumanRoleAssignmentBase" abstract="true" block=""/>
1251
1252
         <xsd:element name="processStakeholders"</pre>
1253
       type="htd:tGenericHumanRoleAssignment" substitutionGroup="genericHumanRole"/>
         <xsd:element name="businessAdministrators"</pre>
1254
1255
       type="htd:tGenericHumanRoleAssignment" substitutionGroup="genericHumanRole"/>
1256
         <xsd:element name="processInitiator" type="htd:tGenericHumanRoleAssignment"</pre>
1257
      substitutionGroup="genericHumanRole"/>
1258
1259
         <!-- element "argument" to be used within "bpel:from" -->
1260
         <xsd:element name="argument" type="tArgument" />
         <xsd:complexType name="tArgument">
1261
1262
           <xsd:complexContent>
1263
             <xsd:extension base="bpel:tExpression">
               <xsd:attribute name="name" type="xsd:NCName" />
1264
1265
             </xsd:extension>
1266
           </xsd:complexContent>
1267
         </xsd:complexType>
```

```
1268
1269
         <!-- attribute "logicalPeopleGroup" to be used within "bpel:from" and
1270
       "bpel:to" -->
1271
         <xsd:attribute name="logicalPeopleGroup" type="xsd:NCName" />
1272
1273
         <!-- attribute "shareComments" to be used within "bpel:process" and
1274
       "bpel:scope" -->
1275
         <xsd:attribute name="shareComments" type="xsd:boolean" />
1276
1277
         <!-- element "peopleActivity" to be used within "bpel:extensionActivity" --
1278
1279
        <xsd:element name="peopleActivity" type="tPeopleActivity" />
1280
         <xsd:complexType name="tPeopleActivity">
1281
           <xsd:complexContent>
1282
             <xsd:extension base="tExtensibleMixedNamespaceElements">
1283
               <xsd:sequence>
1284
                 <xsd:element ref="bpel:targets" minOccurs="0" />
1285
                 <xsd:element ref="bpel:sources" minOccurs="0" />
1286
                 <xsd:choice>
1287
                   <xsd:element ref="htd:task" />
1288
                   <xsd:element ref="localTask" />
1289
                   <xsd:element ref="remoteTask" />
1290
                   <xsd:element ref="htd:notification" />
1291
                   <xsd:element ref="localNotification" />
1292
                   <xsd:element ref="remoteNotification" />
1293
                 </xsd:choice>
1294
                 <xsd:element ref="scheduledActions" minOccurs="0" />
1295
                 <xsd:element ref="toParts" minOccurs="0" />
1296
                 <xsd:element ref="fromParts" minOccurs="0" />
1297
                 <xsd:element ref="attachmentPropagation" minOccurs="0" />
1298
                 <xsd:any namespace="##other" processContents="lax"</pre>
1299
                   minOccurs="0" maxOccurs="unbounded" />
1300
               </xsd:sequence>
1301
               <xsd:attribute name="name" type="xsd:NCName" />
1302
               <xsd:attribute name="suppressJoinFailure" type="tBoolean"</pre>
1303
                use="optional" />
1304
               <xsd:attribute name="inputVariable" type="xsd:QName" />
1305
               <xsd:attribute name="outputVariable" type="xsd:QName" />
1306
               <xsd:attribute name="isSkipable" type="tBoolean"</pre>
1307
                use="optional" default="no" />
1308
               <xsd:attribute name="dontShareComments" type="tBoolean"</pre>
1309
                 use="optional" default="no" />
1310
             </xsd:extension>
1311
           </xsd:complexContent>
1312
         </xsd:complexType>
1313
         <xsd:complexType name="tOverridableTaskElements">
1314
           <xsd:complexContent>
1315
             <xsd:extension base="tExtensibleMixedNamespaceElements">
1316
               <xsd:sequence>
1317
                 <xsd:element ref="htd:priority" minOccurs="0" />
1318
                 <xsd:element ref="htd:peopleAssignments" minOccurs="0" />
1319
               </xsd:sequence>
1320
             </xsd:extension>
1321
           </xsd:complexContent>
1322
         </xsd:complexType>
1323
         <xsd:element name="localTask" type="tLocalTask" />
1324
         <xsd:complexType name="tLocalTask">
1325
           <xsd:complexContent>
```

1326	<pre><xsd:extension base="tOverridableTaskElements"></xsd:extension></pre>
1327	<xsd:attribute <="" name="reference" th="" type="xsd:QName"></xsd:attribute>
1328	use="required" />
1329	
1330	
1331	
1332	<pre><xsd:element name="remoteTask" type="tRemoteTask"></xsd:element></pre>
1333	<pre><xsd:complextype name="tRemoteTask"></xsd:complextype></pre>
1334	<pre><xsd:complexcontent></xsd:complexcontent></pre>
1335	<restension base="t0verridableTaskElements"></restension>
1336	<pre><xsd:attribute <="" name="partnerLink" pre="" type="xsd:NCName"></xsd:attribute></pre>
1337	use="required" />
1338	<rpre><xsd:attribute <="" name="operation" pre="" type="xsd:NCName"></xsd:attribute></rpre>
1339	use="required" />
1340	<pre><xsd:attribute name="responseOperation" type="xsd:NCName"></xsd:attribute></pre>
1341	<pre> <pre> <pre> </pre></pre></pre>
1342	
1343	
1344	<pre><xsd:complextype name="t0verridableNotificationElements"></xsd:complextype></pre>
1345	<xsd:complexcontent></xsd:complexcontent>
1346	<pre><xsd:extension base="tExtensibleMixedNamespaceElements"></xsd:extension></pre>
1347	<pre><xsd:sequence></xsd:sequence></pre>
1348	<pre><xsd:element minoccurs="0" ref="htd:priority"></xsd:element></pre>
1349	<xsd:element minoccurs="0" ref="htd:peopleAssignments"></xsd:element>
1350	
1351	
1352	
1353	
1354	<pre><xsd:element name="localNotification" type="tLocalNotification"></xsd:element></pre>
1355	<pre><xsd:complextype name="tLocalNotification"></xsd:complextype></pre>
1356	<xsd:complexcontent></xsd:complexcontent>
1357	<pre><xsd:extension base="t0verridableNotificationElements"></xsd:extension></pre>
1358	<pre><xsd:attribute <="" name="reference" pre="" type="xsd:QName"></xsd:attribute></pre>
1359	use="required" />
1360	
1361	
1362	
1363	
	<pre><xsd:element name="remoteNotification" type="tRemoteNotification"></xsd:element></pre>
1364	<rpre><xsd:complextype name="tRemoteNotification"></xsd:complextype></rpre>
1365	<xsd:complexcontent></xsd:complexcontent>
1366	<pre><xsd:extension base="t0verridableNotificationElements"></xsd:extension></pre>
1367	
	<pre><xsd:attribute <="" name="partnerLink" pre="" type="xsd:NCName"></xsd:attribute></pre>
1368	use="required" />
1369	<pre><xsd:attribute <="" name="operation" pre="" type="xsd:NCName"></xsd:attribute></pre>
1370	use="required" />
1371	
1372	
1373	
1374	<pre><xsd:element name="scheduledActions" type="tScheduledActions"></xsd:element></pre>
1375	<pre><xsd:complextype name="tScheduledActions"></xsd:complextype></pre>
1376	<xsd:complexcontent></xsd:complexcontent>
1377	<pre><xsd:extension base="tExtensibleElements"></xsd:extension></pre>
1378	<pre><xsd:sequence></xsd:sequence></pre>
1379	<re><rsd:element <="" name="deferActivation" pre=""></rsd:element></re>
1380	type="tScheduledActionsDetails" minOccurs="0" />
1381	<re><rsd:element <="" name="expiration" pre=""></rsd:element></re>
1382	type="tScheduledActionsDetails" minOccurs="0" />
1383	

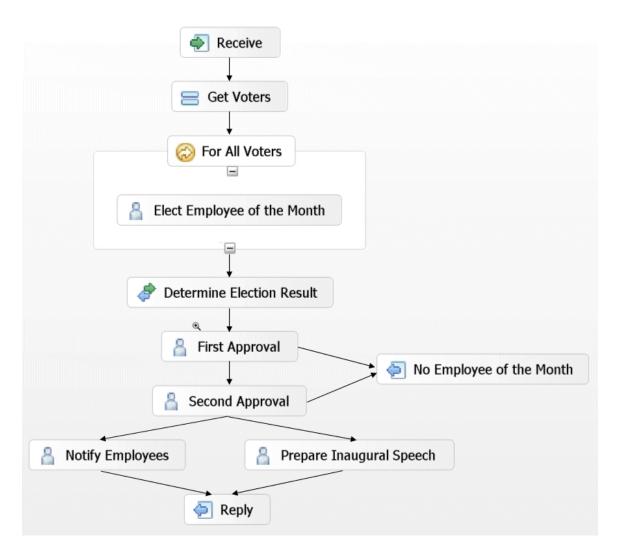
1384	
1385	
1386	
1387	<pre><xsd:complextype name="tScheduledActionsDetails"></xsd:complextype></pre>
1388	<xsd:complexcontent></xsd:complexcontent>
1389	<pre><xsd:extension base="tExtensibleElements"></xsd:extension></pre>
1390	<pre><xsd:sequence></xsd:sequence></pre>
1391	<xsd:sequence></xsd:sequence>
1392	
	<pre><xsd:element name="for" type="bpel:tDuration-expr"></xsd:element></pre>
1393	<pre><xsd:element name="until" type="bpel:tDeadline-expr"></xsd:element></pre>
1394	
1395	
1396	sd:extension>
1397	
1398	
1399	<pre><xsd:element name="fromParts" type="tFromParts"></xsd:element></pre>
1400	<re><xsd:complextype name="tFromParts"></xsd:complextype></re>
1401	<pre><xsd:complexcontent></xsd:complexcontent></pre>
1402	<rpre><rsd:extension base="tExtensibleElements"></rsd:extension></rpre>
1403	<xsd:sequence></xsd:sequence>
1404	<pre><xsd:element maxoccurs="unbounded" ref="fromPart"></xsd:element></pre>
1405	
1406	
1407	
1408	
1409	<pre><xsd:element name="fromPart" type="tFromPart"></xsd:element></pre>
1410	<pre><xsd:complextype name="tFromPart"></xsd:complextype></pre>
1411	<rsd:complexcontent></rsd:complexcontent>
1412	<rpre><xsd:extension base="tExtensibleElements"></xsd:extension></rpre>
1413	<xsd:attribute name="part" type="xsd:NCName" use="required"></xsd:attribute>
1414	<pre><xsd:attribute <="" name="toVariable" pre="" type="bpel:BPELVariableName"></xsd:attribute></pre>
1415	use="required" />
1416	
1417	
1418	
1419	<pre><xsd:element name="toParts" type="tToParts"></xsd:element></pre>
1420	<pre><xsd:complextype name="tToParts"></xsd:complextype></pre>
1421	<pre><xsd:complexcontent></xsd:complexcontent></pre>
1422	<pre><xsd:extension base="tExtensibleElements"></xsd:extension></pre>
1423	<xsd:sequence></xsd:sequence>
1424	<pre><xsd:element maxoccurs="unbounded" ref="toPart"></xsd:element></pre>
1425	
1426	
1427	
1428	
1429	<pre><xsd:element name="toPart" type="tToPart"></xsd:element></pre>
1430	<pre><xsd:complextype name="tToPart"></xsd:complextype></pre>
1431	<pre><xsd:complexcontent></xsd:complexcontent></pre>
1432	<pre><xsd:extension base="tExtensibleElements"></xsd:extension></pre>
1433	<pre><xsd:attribute name="part" type="xsd:NCName" use="required"></xsd:attribute></pre>
1434	<pre><xsd:attribute <="" name="fromVariable" pre=""></xsd:attribute></pre>
1435	type="bpel:BPELVariableName" use="required" />
1436	
1437	
1438	
1439	<pre><xsd:element <="" name="attachmentPropagation" pre=""></xsd:element></pre>
1440	type="tAttachmentPropagation" />
1441	<pre><xsd:complextype name="tAttachmentPropagation"></xsd:complextype></pre>

4 4 4 0	
1442	<pre><xsd:complexcontent></xsd:complexcontent></pre>
1443	<pre><xsd:extension base="tExtensibleElements"></xsd:extension></pre>
1444	<pre><xsd:attribute <="" name="fromProcess" pre="" type="tFromProcess"></xsd:attribute></pre>
1445	default="all" />
1446	<xsd:attribute <="" name="toProcess" th="" type="tToProcess"></xsd:attribute>
1447	default="newOnly" />
1448	
1449	
1450	
1451	<pre><xsd:simpletype name="tFromProcess"></xsd:simpletype></pre>
1452	<pre><xsd:restriction base="xsd:string"></xsd:restriction></pre>
1453	<rpre><rsd:enumeration value="all"></rsd:enumeration></rpre>
1454	<rpre><rsd:enumeration value="none"></rsd:enumeration></rpre>
1455	
1456	
1457	<pre><xsd:simpletype name="tToProcess"></xsd:simpletype></pre>
1458	<xsd:restriction base="xsd:string"></xsd:restriction>
1459	<rpre><xsd:enumeration value="all"></xsd:enumeration></rpre>
1460	<xsd:enumeration value="newOnly"></xsd:enumeration>
1461	<rpre><rsd:enumeration value="none"></rsd:enumeration></rpre>
1462	
1463	
1464	
1465	miscellaneous helper elements and types
1466	<xsd:simpletype name="tBoolean"></xsd:simpletype>
1467	<pre><xsd:restriction base="xsd:string"></xsd:restriction></pre>
1468	<xsd:enumeration value="yes"></xsd:enumeration>
1469	<rpre><xsd:enumeration value="no"></xsd:enumeration></rpre>
1470	
1471	
1472	
1473	

1474 **D. Sample**

1475 This appendix contains a sample that outlines the basic concepts of this specification. The sample

1476 process implements the election of the "Employee of the month" in a fictious company. The structure of 1477 the business process is shown in the figure below:



1478

1479 The process is started and as a first step, the people are determined that qualify as voters for the

1480 "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.

1489 **D.1 BPEL Definition**

```
1490
      <?xml version="1.0" encoding="UTF-8"?>
1491
      <!--
1492
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1493
       -->
1494
      cess name="EmployeeOfTheMonthProcess"
1495
        targetNamespace="http://www.example.com"
1496
        xmlns:tns="http://www.example.com"
1497
         xmlns:hr="http://www.example.com/approval"
1498
         xmlns:el="http://www.example.com/election"
1499
        xmlns:ty="http://www.example.com/types"
1500
        xmlns:ta="http://www.example.com/tasks"
1501
        xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
1502
        xmlns:b4p="http://docs.oasis-open.org/ns/bpel4people/bpel4people/200803"
1503
        xmlns:htd="http://docs.oasis-open.org/ns/bpel4people/ws-humantask/200803"
1504
         xmlns:htt="http://docs.oasis-open.org/ns/bpel4people/ws-
1505
      humantask/types/200803"
1506
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1507
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1508
         xsi:schemaLocation="http://docs.oasis-
1509
      open.org/ns/bpel4people/bpel4people/200803 ../../xml/bpel4people.xsd
1510
      http://docs.oasis-open.org/ns/bpe14people/ws-humantask/200803 ../../xml/ws-
1511
      humantask.xsd http://docs.oasis-open.org/ns/bpel4people/ws-
1512
      humantask/types/200803 ../../xml/ws-humantask-types.xsd">
1513
1514
         <documentation>
1515
           Example for BPEL4People 1.1 - WS-BPEL 2.0 Process with BPEL4People
1516
      Extensions
1517
         </documentation>
1518
1519
         <b4p:humanInteractions>
1520
1521
           <htd:logicalPeopleGroups>
1522
1523
             <htd:logicalPeopleGroup name="voters">
1524
               <htd:documentation xml:lang="en-US">
1525
                 The group entitled to vote the employee of the month for the
1526
                 given region.
1527
               </htd:documentation>
1528
               <htd:parameter name="region" type="xsd:string" />
1529
             </htd:logicalPeopleGroup>
1530
1531
             <htd:logicalPeopleGroup name="approvers">
1532
               <htd:documentation xml:lang="en-US">
1533
                 The group entitled to approve the elected employee of the
1534
                 month for the given region.
1535
               </htd:documentation>
1536
               <htd:parameter name="region" type="xsd:string" />
1537
             </htd:logicalPeopleGroup>
1538
1539
             <htd:logicalPeopleGroup name="employees">
               <htd:documentation xml:lang="en-US">
1540
1541
                 The group of employees to be notified about the election
1542
                 result of the employee of the month election for the given
1543
                region.
1544
               </htd:documentation>
1545
               <htd:parameter name="region" type="xsd:string" />
```

```
1546
             </htd:logicalPeopleGroup>
1547
1548
             <htd:logicalPeopleGroup name="regionalElectionCommittee">
1549
               <htd:documentation xml:lang="en-US">
1550
                 The group who is in charge for the election of the
1551
                 employee of the month election for the given region.
1552
               </htd:documentation>
               <htd:parameter name="region" type="xsd:string" />
1553
1554
             </htd:logicalPeopleGroup>
1555
1556
           </htd:logicalPeopleGroups>
1557
1558
           <htd:tasks>
1559
             <htd:task name="approveEmployeeOfTheMonth">
1560
               <htd:documentation xml:lang="en-US">
                  The reusable definition of the task used to approve the
1561
1562
                  election of the employee of the month.
1563
               </htd:documentation>
1564
               <htd:interface operation="approve" portType="hr:approvalPT"/>
1565
               <htd:peopleAssignments>
1566
                 <htd:potentialOwners>
1567
                   <htd:from logicalPeopleGroup="approvers">
1568
                     <!-- variables used here need to be defined on the
1569
                          enclosing scope or above -->
1570
                     <htd:argument name="region">
1571
                       $electionRequest/region
1572
                     </htd:argument>
1573
                   </htd:from>
1574
                 </htd:potentialOwners>
1575
               </htd:peopleAssignments>
1576
               <htd:presentationElements/>
1577
             </htd:task>
1578
           </htd:tasks>
1579
1580
         </bap:humanInteractions>
1581
1582
         <b4p:peopleAssignments>
1583
1584
           <b4p:processStakeholders>
1585
             <htd:from logicalPeopleGroup="regionalElectionCommittee">
1586
               <htd:argument name="region">
1587
                 $electionRequest/region
1588
               </htd:argument>
1589
             </htd:from>
1590
           </bd></bd></
1591
1592
           <b4p:businessAdministrators>
1593
             <htd:from>
1594
               <htd:literal>
1595
                 <htt:organizationalEntity>
1596
                   <htt:user>Peter</htt:user>
1597
                   <htt:user>Paul</htt:user>
1598
                   <htt:user>Mary</htt:user>
1599
                 </htt:organizationalEntity>
1600
               </htd:literal>
1601
             </htd:from>
1602
           </b4p:businessAdministrators>
1603
```

```
1604
         </b4p:peopleAssignments>
1605
1606
         <extensions>
1607
           <extension
1608
             namespace="http://docs.oasis-
1609
       open.org/ns/bpel4people/bpel4people/200803"
1610
             mustUnderstand="yes"/>
           <extension
1611
1612
             namespace="http://docs.oasis-open.org/ns/bpel4people/ws-
1613
       humantask/200803"
1614
             mustUnderstand="yes"/>
1615
         </extensions>
1616
1617
         <import
1618
           importType="http://www.w3.org/2001/XMLSchema"
1619
           namespace="http://www.example.com/types"/>
1620
         <import
1621
           importType="http://www.example.org/WS-HT"
1622
           namespace="http://www.example.com/tasks"/>
1623
         <import
1624
           importType="http://schemas.xmlsoap.org/wsdl/"
1625
           namespace="http://www.example.com/election"
1626
           location="bpel4people-example-election.wsdl"/>
1627
         <import
1628
           importType="http://schemas.xmlsoap.org/wsdl/"
1629
           namespace="http://www.example.com/approval"
1630
           location="bpel4people-example-approval.wsdl"/>
1631
1632
         <partnerLinks>
1633
           <partnerLink partnerLinkType="electionPLT"</pre>
1634
                        name="electionPL"/>
1635
         </partnerLinks>
1636
1637
         <variables>
1638
           <variable name="candidates" type="htt:users"/>
1639
           <variable name="voters" type="htd:tOrganizationalEntity"/>
1640
           <variable name="electionRequest" type="ty:electionRequestData"/>
1641
           <variable name="electionResult" type="ty:electionResultData"/>
1642
           <variable name="decision" type="xsd:boolean"/>
1643
           <variable name="speech" type="ty:document"/>
1644
         </variables>
1645
1646
         <sequence>
1647
1648
           <receive partnerLink="electionPL"
1649
                    portType="el:electionPT"
1650
                    operation="elect"
1651
                    variable="electionRequest"
1652
                    createInstance="yes"/>
1653
1654
           <assign name="getVoters">
1655
             <copy>
1656
               <from>$electionRequests/candidates</from>
1657
               <to variable="candidates"/>
1658
             </copy>
1659
             <copy>
1660
               <from b4p:logicalPeopleGroup="voters">
1661
                 <b4p:argument name="region">
```

```
1662
                    $electionRequest/region
1663
                 </b4p:argument>
1664
               </from>
1665
               <to variable="voters" />
1666
             </copy>
1667
           </assign>
1668
1669
           <forEach counterName="i" parallel="yes">
1670
             <startCounterValue> 1 </startCounterValue>
1671
             <finalCounterValue>
1672
               count($voters/users/user)
1673
             </finalCounterValue>
1674
1675
             <scope>
1676
               <variables>
1677
                   <variable name="vote" type="htt:user"/>
1678
               </variables>
1679
1680
               <sequence>
1681
                 <!-- Constellation 1 -->
1682
                 <extensionActivity>
1683
                    <b4p:peopleActivity name="electEmployeeOfTheMonth"
1684
                                         inputVariable="candidates"
1685
                                         outputVariable="vote"
1686
                                         isSkipable="yes">
1687
                      <htd:task name="votingTask">
1688
                        <htd:interface operation="vote"
                                       portType="el:votingPT"/>
1689
1690
                        <htd:peopleAssignments>
1691
                          <htd:potentialOwners>
1692
                            <htd:from>$voters/users/user[i]</htd:from>
1693
                          </htd:potentialOwners>
1694
                        </htd:peopleAssignments>
1695
                        <htd:presentationElements/>
1696
                      </htd:task>
1697
                      <b4p:scheduledActions>
1698
                        <b4p:expiration>
1699
                          <b4p:documentation xml:lang="en-US">
1700
                            This people activity expires when not completed
1701
                            within 2 days after having been activated.
1702
                          </b4p:documentation>
1703
                          <b4p:for>P2D</b4p:for>
1704
                        </b4p:expiration>
1705
                      </b4p:scheduledActions>
1706
                    </b4p:peopleActivity>
1707
                 </extensionActivity>
1708
1709
                 <assign>
1710
                   <copy>
1711
                      <from>$vote</from>
1712
                      <to>$electionResult/votes[i]</to>
1713
                    </copy>
1714
                 </assign>
1715
1716
               </sequence>
1717
             </scope>
1718
           </forEach>
1719
```

```
1720
           <!-- Might be Constellation 5 - standard WS-BPEL 2.0 invoke -->
1721
           <!--
1722
           <invoke name="determineElectionResult" partnerLink="..." operation="..."</pre>
1723
       />
1724
           -->
1725
1726
           <!-- Constellation 2 -->
1727
           <extensionActivity>
1728
             <b4p:peopleActivity name="firstApproval"
1729
                                  inputVariable="electionResult"
1730
                                  outputVariable="decision">
1731
               <b4p:localTask reference="tns:approveEmployeeOfTheMonth"/>
1732
             </b4p:peopleActivity>
1733
           </extensionActivity>
1734
1735
           <!-- Constellation 2 with override specifications -->
1736
           <extensionActivity>
1737
             <b4p:peopleActivity name="secondApproval"
1738
                                  inputVariable="electionResult"
1739
                                  outputVariable="decision">
1740
               <b4p:localTask reference="tns:approveEmployeeOfTheMonth">
1741
                 <htd:peopleAssignments>
1742
                   <htd:excludedOwners>
1743
                     <htd:from>
1744
                       b4p:getActualOwner("tns:firstApproval")
1745
                     </htd:from>
                   </htd:excludedOwners>
1746
                 </htd:peopleAssignments>
1747
1748
               </b4p:localTask>
1749
             </b4p:peopleActivity>
1750
           </extensionActivity>
1751
1752
           <!-- Constellation 3 -->
1753
           <extensionActivity>
1754
             <b4p:peopleActivity name="notifyEmployees"
1755
                                  inputVariable="electionResult">
1756
               <b4p:localNotification reference="ta:employeeBroadcast"/>
1757
               <!-- notification is not defined as part of this document,
                    but within a separate one
1758
1759
               -->
1760
             </b4p:peopleActivity>
1761
           </extensionActivity>
1762
1763
           <!-- Constellation 4 -->
1764
           <extensionActivity>
             <b4p:peopleActivity name="prepareInauguralSpeech"
1765
1766
                                  inputVariable="electionResult"
1767
                                  outputVariable="speech"
1768
                                  isSkipable="no">
1769
               <b4p:remoteTask partnerLink="author"
1770
                                operation="prepareSpeech"
1771
                                responseOperation="receiveSpeech">
1772
                 <htd:priority>0</htd:priority> <!-- assign highest priority -->
1773
                 <htd:peopleAssignments>
1774
                     <htd:potentialOwners>
1775
                       <htd:from>$electionResult/winner</htd:from>
1776
                     </htd:potentialOwners>
1777
                 </htd:peopleAssignments>
```

```
1778 </bdp:remoteTask>
1779 </bdp:peopleActivity>
1780 </extensionActivity>
1781
1782 </sequence>
1783
1784 </process>
```

1785 D.2 WSDL Definitions

```
1786
      <?xml version="1.0" encoding="UTF-8"?>
1787
      <!--
1788
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1789
       -->
      <wsdl:definitions
1790
1791
         xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1792
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1793
         xmlns:tns="http://www.example.com/approval"
1794
         targetNamespace="http://www.example.com/approval">
1795
1796
         <wsdl:documentation>
1797
           Example for BPEL4People 1.1 - PeopleActivity Interface Definition
1798
         </wsdl:documentation>
1799
1800
         <!-- Messages -->
1801
         <wsdl:message name="approvalInput">
1802
           <wsdl:part name="parameters" type="xsd:string" />
1803
         </wsdl:message>
1804
         <wsdl:message name="approvalOutput">
1805
           <wsdl:part name="parameters" type="xsd:string" />
1806
         </wsdl:message>
1807
1808
         <!-- Port Type -->
1809
         <wsdl:portType name="approvalPT">
1810
           <wsdl:operation name="approve">
1811
             <wsdl:input message="tns:approvalInput" />
1812
             <wsdl:output message="tns:approvalOutput" />
1813
           </wsdl:operation>
1814
         </wsdl:portType>
1815
1816
      </wsdl:definitions>
1817
1818
       <?xml version="1.0" encoding="UTF-8"?>
1819
      <!--
1820
        Copyright (c) OASIS Open 2009. All Rights Reserved.
1821
       -->
1822
       <wsdl:definitions
1823
         xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1824
         xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1825
         xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
1826
         xmlns:tns="http://www.example.com/election"
1827
         targetNamespace="http://www.example.com/election">
1828
1829
         <wsdl:documentation>
1830
           Example for BPEL4People 1.1 - PeopleActivity Interface Definition
1831
         </wsdl:documentation>
1832
```

1833	WS-BPEL 2.0 Partner Link Type		
1834	<plnk:partnerlinktype name="electionPLT"></plnk:partnerlinktype>		
1835	<pre><plnk:role name="electionService" porttype="tns:electionPT"></plnk:role></pre>		
1836			
1837			
1838	Messages		
1839	<pre><wsdl:message name="electionInput"></wsdl:message></pre>		
1840	<pre><wsdl:part name="parameters" type="xsd:string"></wsdl:part></pre>		
1841			
1842	<pre><wsdl:message name="votingInput"></wsdl:message></pre>		
1843	<pre><wsdl:part name="parameters" type="xsd:string"></wsdl:part></pre>		
1844			
1845	() () () () () () () () () () () () () (
1846	Port Types		
1847	<pre><wsdl:porttype name="electionPT"></wsdl:porttype></pre>		
1848	<pre><wsdl:operation name="elect"></wsdl:operation></pre>		
1849	<pre><wsdl:operation name="erect"></wsdl:operation></pre>		
1850			
1851			
1852	<wsdl:porttype name="votingPT"></wsdl:porttype>		
1853	<pre><wsdl:operation name="vote"></wsdl:operation></pre>		
1854	<pre><wsdl:input message="tns:votingInput"></wsdl:input></pre>		
1855			
1856			
1857			
1858			

E. Acknowledgements 1859

1860 The following individuals have participated in the creation of this specification and are gratefully 1861 acknowledged:

Members of the BPEL4People Technical Committee:

- 1862 1863 1864 Phillip Allen, Microsoft Corporation 1865 Ashish Agrawal, Adobe Systems 1866 Mike Amend, BEA Systems, Inc. Stefan Baeuerle, SAP AG 1867 1868 Charlton Barreto, Adobe Systems 1869 Justin Brunt, TIBCO Software Inc. 1870 Martin Chapman, Oracle Corporation 1871 Luc Clément, Active Endpoints, Inc. 1872 Manoj Das, Oracle Corporation 1873 Alireza Farhoush, TIBCO Software Inc. 1874 Mark Ford, Active Endpoints, Inc. 1875 Sabine Holz, SAP AG 1876 Dave Ings, IBM 1877 Gershon Janssen, Individual 1878 Diane Jordan, IBM 1879 Anish Karmarkar, Oracle Corporation 1880 Ulrich Keil, SAP AG 1881 Oliver Kieselbach, SAP AG 1882 Matthias Kloppmann, IBM 1883 Dieter König, IBM 1884 Marita Kruempelmann, SAP AG 1885 Frank Leymann, IBM 1886 Mark Little, Red Hat 1887 Alexander Malek, Microsoft Corporation 1888 Ashok Malhotra, Oracle Corporation 1889 Mike Marin, IBM Vinkesh Mehta, Deloitte Consulting LLP 1890 1891 Jeff Mischkinsky, Oracle Corporation 1892 Ralf Mueller, Oracle Corporation 1893 Krasimir Nedkov, SAP AG 1894 Benjamin Notheis, SAP AG 1895 Michael Pellegrini, Active Endpoints, Inc. 1896 Hannah Petereit, SAP AG
- 1897 Gerhard Pfau, IBM
- 1898 Karsten Ploesser, SAP AG

1899	Ravi Rangaswamy, Oracle Corporation
1900	Alan Rickayzen, SAP AG
1901	Michael Rowley, BEA Systems, Inc.
1902	Ron Ten-Hove, Sun Microsystems
1903	Ivana Trickovic, SAP AG
1904	Alessandro Triglia, OSS Nokalva
1905	Claus von Riegen, SAP AG
1906	Peter Walker, Sun Microsystems
1907	Franz Weber, SAP AG
1908	Prasad Yendluri, Software AG, Inc.
1909	
1910	BPEL4People 1.0 Specification Contributors:
1911	Ashish Agrawal, Adobe
1912	Mike Amend, BEA
1913	Manoj Das, Oracle
1914	Mark Ford, Active Endpoints
1915	Chris Keller, Active Endpoints
1916	Matthias Kloppmann, IBM
1917	Dieter König, IBM
1918	Frank Leymann, IBM
1919	Ralf Müller, Oracle
1920	Gerhard Pfau, IBM
1921	Karsten Plösser, SAP
1922	Ravi Rangaswamy, Oracle
1923	Alan Rickayzen, SAP
1924	Michael Rowley, BEA
1925	Patrick Schmidt, SAP
1926	Ivana Trickovic, SAP
1927	Alex Yiu, Oracle
1928	Matthias Zeller, Adobe
1929	
1930	In addition, the following individuals have provide

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

1933 Weber, and Eric Wittmann.

1934 **F. Non-Normative Text**

1935 **G. Revision History**

1936 [optional; should not be included in OASIS Standards]

1937

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

Revision	Date	Editor	Changes Made
CD-01-rev-3	2008-12-03	Ralf Mueller	Resolution for Issue #16 applied to sections 1 – 6
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

1938