Appendix A: E-Mail Voting Process BPEL4WS

This appendix provides the complete BPEL4WS code for the example BPMN business process that is described in the section entitled “BPMN by Example” on page 201.

```
<definitions
    targetNamespace="http://www.website.com"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <message name="processDataMessage">
    <part name="NumIssues" type="xsd:integer"/>
    <part name="NoMajority" type="xsd:boolean"/>
    <part name="VotedOnce" type="xsd:boolean"/>
    <part name="NumVoted" type="xsd:integer"/>
    <part name="VotersWarned" type="xsd:boolean"/>
    <part name="LoopCounter" type="xsd:integer"/>
  </message>
    <!--processDataMessage will be received with the following parts:
        NoMajority (set to false)
        VotedOnce (set to false)
        NumVoted (set to false)
        VotersWarned (set to false)
        LoopCounter (set to 0)
        starting message every Friday is not shown here.-->
</definitions>

<!-- The Main Process -->
<process name="EMailVotingProcess">
  <variables>
    <variable name="processData" messageType="processDataMessage"/>
    <!--processDataMessage will be received with the following parts:
        NumIssues (set to the number of unresolved Issues)
        NoMajority (set to false)
        VotedOnce (set to false)
        NumVoted (set to false)
        VotersWarned (set to false)
        LoopCounter (set to 0)
        starting message every Friday is not shown here.-->
  </variables>
  <sequence>
    <!--This starts the beginning of the Process. The process that sends the
        starting message every Friday is not shown here.-->
    <receive partnerLink="Internal" portType="tns:processPort"
            operation="receiveIssueList" variable="processData" createInstance="Yes"/>
    <invoke name="ReviewIssueList" partnerLink="Internal" portType="tns:internalPort"
            operation="sendIssueList" inputVariable="processData"
            outputVariable="processData"/>
    <switch name="AnyIssuesReady">
```

Copyright © 2004, BPMI.org All Rights Reserved

```
      <!--name="Yes" -->
      <case condition="bpws:getVariableProperty(ProcessData,NumIssues)>0">
        <!-- A chunk of this process is separated into a derived process so that
             it can be called from a complex loop. -->
        <invoke name="Discussion_Cycle_Derived_Process" partnerLink="Internal"
               portType="tns:processPort" operation="call_Discussion_Cycle_Derived_Process"
                inputVariable="processData" outputVariable="processData"/>
      </case>
      <!--name="No" -->
      <otherwise>
        <!--This is one of the two ways to the end of the Process.-->
        <empty/>
      </otherwise>
    </switch>
  </sequence>

<!-- A Derived Process -->
<process name="Discussion_Cycle_Derived_Process">
  <variables>
    <variable name="processData" messageType="processDataMessage"/>
    <variable name="Discussion_Cycle_loopCounter" messageType="loopCounterMessage"/>
  </variables>
  <sequence>
    <receive partnerLink="Internal" portType="tns:processPort"
            operation="call_Discussion_Cycle_Derived_Process" variable="processData"
             createInstance="Yes"/>
    <!--The first Sub-Process has a loop condition, so it is within a while-->
    <assign name="Discussion_Cycle_initialize_loopCounter">
      <copy>
        <from expression="0"/>
        <to variable="Discussion_Cycle_loopCounter" part="loopCounter" />
      </copy>
    </assign>
    <!--Since the TestTime is "After" the Sub-Process has to be performed before the
        while-->
    <invoke name="Discussion_Cycle" partnerLink="Internal"
           portType="tns:processPort operation="call_Discussion_Cycle"
           inputVariable="processData" outputVariable="processData"/>
    <while condition="bpws:getVariableProperty(ProcessData,DiscussionOver)=false">
      <!--This calls the first Sub-Process-->
      <sequence>
        <invoke process="Discussion_Cycle" partnerLink="Internal"
                portType="tns:processPort operation="call_Discussion_Cycle"
                inputVariable="processData" outputVariable="processData"/>
        <assign>
          <copy>
            <from expression=
                  "bpws:getVariableProperty(Discussion_Cycle_loopCounter,LoopCounter)+1"/>
            <to variable="Discussion_Cycle_loopCounter" part="LoopCounter"/>
          </copy>
        </assign>
      </sequence>
    </while>
```


Copyright © 2004, BPMI.org All Rights Reserved

```
    <!--This calls the first another derived process to handle the rest of the
        work-->
    <invoke name="Announce_Issues_Derived_Process" partnerLink="Internal"
            portType="tns:processPort" operation="call_Announce_Issues_Derived_Process"
            inputVariable="processData"  outputVariable="processData"/>
  </sequence>
</process>

</process>
<!-- A Derived Process -->
<process name="Announce_Issues_Derived_Process">
  <!-- This starts the middle section of the process. -->
  <variables>
    <variable name="processData" messageType="processDataMessage"/>
  </variables>
  <sequence>
    <receive partnerLink="Internal" portType="tns:processPort"
            operation="call_Announce_Issues_Derived_Process" variable="processData"
             createInstance="Yes"/>
    <invoke name="AnnounceIssuesforVote" partnerLink="WGVoter" portType="tns:emailPort"
           operation="sendVoteAnnouncement" inputVariable="processData"/>
    <invoke name="Collect_Votes_Derived_Process" partnerLink="Internal"
            portType="tns:processPort" operation="call_Collect_Votes_Derived_Process"
            inputVariable="processData"  outputVariable="processData"/>
    <reply partnerLink="Internal" portType="tns:processPort"
          operation="call_Announce_Issues_Derived_Process"
          variable="processData" createInstance="Yes"/>
  </sequence>
</process>

<!-- A Derived Process -->
<process name="Collect_Votes_Derived_Process">
  <!--this calls the second Sub-Process. After the Collect Votes Sub-Process
       times out, the rest of the process will be in the fault handler
       of that process. Calls from there will loop back into other processes.-->
  <variables>
    <variable name="processData" messageType="processDataMessage"/>
  </variables>
  <sequence>
    <receive partnerLink="Internal" portType="tns:processPort"
            operation="call_Collect_Votes_Derived_Process" variable="processData"
            createInstance="Yes"/>
    <invoke name="Collect_Votes" partnerLink="Internal" portType="tns:processPort"
           operation="call_Collect_Votes" inputVariable="processData"
            outputVariable="processData"/>
    <reply partnerLink="Internal" portType="tns:processPort"
         operation="call_Collect_Votes_Derived_Process" variable="processData"
         createInstance="Yes"/>
  </sequence>
</process>
```

Copyright © 2004, BPMI.org All Rights Reserved

```
<!-- A Derived Process -->
<process name="Issues_wo_Majority_Derived_Process">
  <variables>
    <variable name="processData" messageType="processDataMessage"/>
  </variables>
  <sequence>
    <receive partnerLink="Internal" portType="tns:processPort"
            operation="call_Issues_wo_Majority_Derived_Process" variable="processData"
             createInstance="Yes"/>
    <switch name="IssueswoMajority">
      <case name="Yes"
           condition="bpws:getVariableProperty(ProcessData,NoMajority)=true">
        <switch name="2ndTime">
          <!-- name="Yes" -->
          <case condition="bpws:getVariableProperty(ProcessData,VotedOnce)=true">
            <!--This is done to do the complex looping situation. -->
            <invoke name="Discussion_Cycle_Derived_Process" partnerLink="Internal"
                   portType="tns:processPort"
                    operation="call_Discussion_Cycle_Derived_Process"
                   inputVariable="processData" outputVariable="processData"/>
          </case>
          <!-- name="No (otherwise)" -->
          <otherwise>
            <sequence>
              <flow>
                <invoke name="ReducetoTwoSolutions" partnerLink="internal"
                       portType="tns:internalPort" operation="sendReceiveSolutions"
                       inputVariable="processData" outputVariable="processData"/>
                <invoke name="EMail Voters that have to Change Votes"
                        partnerLink="WGVoter" portType="tns:emailPort"
                        operation="sendVoteWarning" inputVariable="processData"/>
              </flow>
              <invoke process="Announce_Issues_Derived_Process" partnerLink="Internal"
                      portType="tns:processPort"
                      operation="call_Announce_Issues_Derived_Process"
                      inputVariable="processData" outputVariable="processData"/>
            </sequence>
          </otherwise>
        </switch>
      </case>
      <otherwise name="Nootherwise">
        <!-- This is one of the two ways to the end of the Process. -->
        <empty/>
      </otherwise>
    </switch>
  </sequence>
</process>

<!-- A User Built Process -->
<process name="Discussion_Cycle">
  <!--This defines the first Sub-Process. -->
  <variables>
    <variable name="processData" messageType="processDataMessage"/>
  </variables>
```


Copyright © 2004, BPMI.org All Rights Reserved

```
<sequence>
  <receive partnerLink="Internal" portType="tns:processPort"
           operation="call_Discussion_Cycle" variable="processData"
           createInstance="Yes"/>
  <invoke name="AnnounceIssuesforDiscussion" partnerLink="WGVoter"
         portType="tns:emailPort" operation="sendDiscussionAnnouncement"
         inputVariable="processData"/>
  <flow>
    <links>
      <link name="CheckCalendarforConferenceCalltoWaituntilThursday9am"/>
      <link name="CheckCalendarforConferenceCalltoEmpty"/>
      <link name="WaituntilThursday9amtoModerateConferenceCallDiscussion"/>
    </links>
    <!-- This is the first of the three paths of the fork. -->
    <scope>
      <invoke name="ModerateEmailDiscussion" partnerLink="internal"
             portType="tns:internalPort" operation="sendDiscussion"
             inputVariable="processData" outputVariable="processData"/>
      <faultHandlers>
        <catch faultName="7Days_Exit">
          <empty/>
        </catch>
      </faultHandlers>
      <eventHandlers>
        <onAlarm for="tns:OneWeek">
          <throw faultName="7Days_Exit"/>
        </catch>
      </eventHandlers>
    </scope>
    <!-- This is the second of the three paths of the fork. -->
    <sequence>
      <wait name="Delay6daysfromDiscussionAnnouncement" for="P6D"/>
      <invoke name="EMailDiscussionDeadlineWarning" partnerLink="WGVoter"
              portType="tns:emailPort" operation="sendDiscussionWarning"
              inputVariable="processData">
      </invoke>
    </sequence>
    <!-- This is the third of the three paths of the fork. -->
    <invoke name="CheckCalendarforConferenceCall" partnerLink="internal"
           portType="tns:internalPort" operation="receiveCallSchedule"
           inputVariable="processData" outputVariable="processData">
      <source linkName="CheckCalendarforConferenceCalltoWaituntilThursday9am"
          transitionCondition="bpws:getVariableProperty(processData,conCall)=true"/>
      <source linkName="CheckCalendarforConferenceCalltoEmpty"
        transitionCondition="not(bpws:getVariableProperty(processData,conCall)=true)"/>
    </invoke>
    <!-- name="Yes" -->
    <wait name="WaituntilThursday9am" for="P6DT9H">
      <target linkName=
             "CheckCalendarforConferenceCalltoWaituntilThursday9am">
      <source linkName="WaituntilThursday9amtoModerateConferenceCallDiscussion"/>
    </wait>
```

Copyright © 2004, BPMI.org All Rights Reserved

```
      <invoke name="ModerateConferenceCallDiscussion" partnerLink="internal"
              portType="tns:internalPort" operation="sendConCall"
              inputVariable="processData" outputVariable="processData">
        <target linkName="WaituntilThursday9amtoModerateConferenceCallDiscussion"/>
      </invoke>
      <!-- name="otherwise" -->
      <empty>
        <target linkName="CheckCalendarforConferenceCalltoEmpty"/>
      </empty>
    </flow>
    <invoke name="EvaluateDiscussionProgress" partnerLink="internal"
          portType="tns:internalPort" operation="receiveDiscussionStatus"
          inputVariable="processData" outputVariable="processData"/>
    <reply partnerLink="Internal" portType="tns:processPort"
           operation="call_Discussion_Cycle" variable="processData"/>
  </sequence>
</process>

<!-- A User Built Process -->
<process name="Collect_Votes">
  <!--This is a process for the E-Mail Voting collection. It consists of an all and a
      timeout event handler. The all will never complete normally since there is an
      infinite loop inside. The timeout is intended to be the normal way of ending the
      process. -->
  <variables>
    <variable name="processData" messageType="processDataMessage"/>
  </variables>
  <sequence>
    <receive partnerLink="Internal" portType="tns:processPort"
           operation="call_Collect_Votes" variable="processData" createInstance="Yes"/>
    <scope>
      <flow>
        <links>
          <link name="Delay6daysfromVoteAnnouncementtoEMailVoteDeadlineWarning"/>
          <link name="CheckCalendarforConferenceCalltoWaituntilThursday9am"/>
          <link name="CheckCalendarforConferenceCalltoEmpty"/>
          <link name="WaituntilThursday9amtoModerateConferenceCallDiscussion"/>
        </links>
        <!--This is the first of the four paths of the fork. -->
        <invoke name="CheckCalendarforConferenceCall" partnerLink="internal"
              portType="tns:internalPort" operation="receiveCallSchedule"
              inputVariable="processData" outputVariable="processData">
          <target linkName="CheckCalendarforConferenceCalltoWaituntilThursday9am"
             transitionCondition="bpws:getVariableProperty(processData,conCall)=true"/>
          <target linkName="CheckCalendarforConferenceCalltoEmpty"
            transitionCondition="not(bpws:getVariableProperty(processData,conCall)=true)"/>
        </invoke>
      <!-- name="Yes" -->
        <wait name="WaituntilThursday9am" for="P6DT9H">
          <source linkName=
                "CheckCalendarforConferenceCalltoWaituntilThursday9am">
          <target linkName="WaituntilThursday9amtoModerateConferenceCallDiscussion"/>
        </wait>
```


Copyright © 2004, BPMI.org All Rights Reserved

```
    <invoke name="ModerateConferenceCallDiscussion" partnerLink="internal"
            portType="tns:internalPort" operation="sendConCall"
            inputVariable="processData" outputVariable="processData">
      <source linkName="WaituntilThursday9amtoModerateConferenceCallDiscussion"/>
    </invoke>
    <!-- name="otherwise" -->
    <empty>
      <source linkName="CheckCalendarforConferenceCalltoEmpty"/>
    </empty>
    <!-- This is the second of the four paths of the fork. -->
    <invoke name="ModerateEMailDiscussion" partnerLink="internal"
            portType="tns:internalPort" operation="sendDiscussion"
            inputVariable="processData" outputVariable="processData"/>
    <!--This is the third of the four paths of the fork.-->
    <wait name="Delay6daysfromVoteAnnouncement" for="P6D">
      <target linkName="Delay6daysfromVoteAnnouncementtoEMailVoteDeadlineWarning"/>
    </wait>
    <invoke name="EMailVoteDeadlineWarning" partnerLink="WGVoter"
            portType="tns:emailPort" operation="sendVoteWarning"
            inputVariable="processData">
      <source linkName="Delay6daysfromVoteAnnouncementtoEMailVoteDeadlineWarning"/>
    </invoke>
    <!--This is the fourth of the four paths of the fork. This branch of the all is
        intended to be an infinite loop that is eventually interrupted by the Time
        Out. This is necessary since any voter can change their vote until the
        deadline. -->
    <while condition="1=0">
      <sequence>
        <receive name="ReceiveVote" partnerLink="WGVoter" portType="tns:emailPort"
                 operation="receiveVote" variable="processData"/>
        <invoke name="IncrementTally" partnerLink="internal"
                portType="tns:internalPort" operation="sendReceiveTotal"
                inputVariable="processData" outputVariable="processData"/>
      </sequence>
    </while>
  </flow>
  <eventHandlers>
    <onAlarm for="P7D">
      <throw faultName="7days_Exit"/>
    </onAlarm>
  </eventHandlers>
  <faultHandlers>
    <catch faultName="7days_Exit">
      <!-- The BPMN Diagram shows that the Timer Intermediate Event connects
           directly to the rest of the Process. Thus, they will show up in
           this activity set. -->
        <sequence>
        <invoke name="PrepareResults" partnerLink="internal"
              portType="tns:internalPort" operation="sendReceiveResults"
              inputVariable="processData" outputVariable="processData"/>
```

Copyright © 2004, BPMI.org All Rights Reserved

```
<flow>
  <invoke name="PostResultsonWebSite" partnerLink="internal"
         portType="tns:internalPort" operation="postVotingResults"
         inputVariable="processData"/>
  <invoke name="EMailResultsofVote" partnerLink="WGVoter"
         portType="tns:emailPort" operation="sendVotingResults"
         inputVariable="processData"/>
</flow>
<switch name="DidEnoughMembersVote">
  <!-- name="No" -->
  <case condition="bpws:getVariableProperty(ProcessData,NumVoted)>
               (.7)*(bpws:getVariableProperty(ProcessData,NumVWGM))">
    <switch name="Havethemembersbeenwarned">
      <!-- name="Yes" -->
      <case condition="bpws:getVariableProperty(ProcessData,
                     VotersWarned)=true">
        <sequence>
          <invoke name="ReducenumberofVotingMembersandRecalculateVote"
                 partnerLink="internal" portType="tns:internalPort"
                 operation="sendReceiveNumVoters" inputVariable="processData"
                 outputVariable="processData"/>
          <!--Some elements of the process were separated into a derived process
               since they would have been repeated. They would have been
               repeated because they are arrived by alternativepaths that do not
               close a set of alternative paths. -->
          <invoke name="Issues_wo_Majority_Derived_Process" partnerLink="Internal"
                 PortType="tns:processPort"
                    operation="call_Issues_wo_Majority_Derived_Process"
                 inputVariable="processData" outputVariable="processData"/>
        </sequence>
      </case>
      <!-- name="No (otherwise)" -->
      <otherwise>
        <sequence>
          <invoke name="ReannounceVotewithwarningtovotingmembers"
                 partnerLink="WGVoter" portType="tns:emailPort"
                 operation="sendReannounceVote" inputVariable="processData"
                 outputVariable="processData"/>
          <invoke name="Collect_Votes_Derived_Process" partnerLink="Internal"
                  portType="tns:processPort"
                 operation="call_Collect_Votes_Derived_Process"
                 inputVariable="processData" outputVariable="processData"/>
        </sequence>
      </otherwise>
    </switch>
  </case>
```


Copyright © 2004, BPMI.org All Rights Reserved

```
          <!-- name="Yes (otherwise)" -->
            <otherwise>
              <!-- Some elements of the process were separated into a derived process
                   since they would have been repeated. They would have been repeated
                   because they are arrived by alternative that do not close a set of
                   alternative paths. -->
              <invoke process="Issues_wo_Majority_Derived_Process" partnerLink="Internal"
                      portType="tns:processPort"
                         operation="call_Issues_wo_Majority_Derived_Process"
                      inputVariable="processData" outputVariable="processData"/>
            </otherwise>
          </switch>
        </sequence>
      </catch>
    </faultHandlers>
  </scope>
  <reply partnerLink="Internal" portType="tns:processPort"
          operation="call_Collect_Votes" variable="processData"/>
 </sequence>
</process>
```

Copyright © 2004, BPMI.org All Rights Reserved

Copyright © 2004, BPMI.org All Rights Reserved