

More BPEL Activities

Objectives

After completing this lesson, you should be able to:

- Perform conditional branching by using an If activity
- Implement parallel processing by using a Flow activity
- Implement non-blocking invocation with a Flow
- Create parallel branches dynamically with a forEach activity
- Implement a Pick activity with an alarm and a timeout
- Execute activities repetitively with a While activity

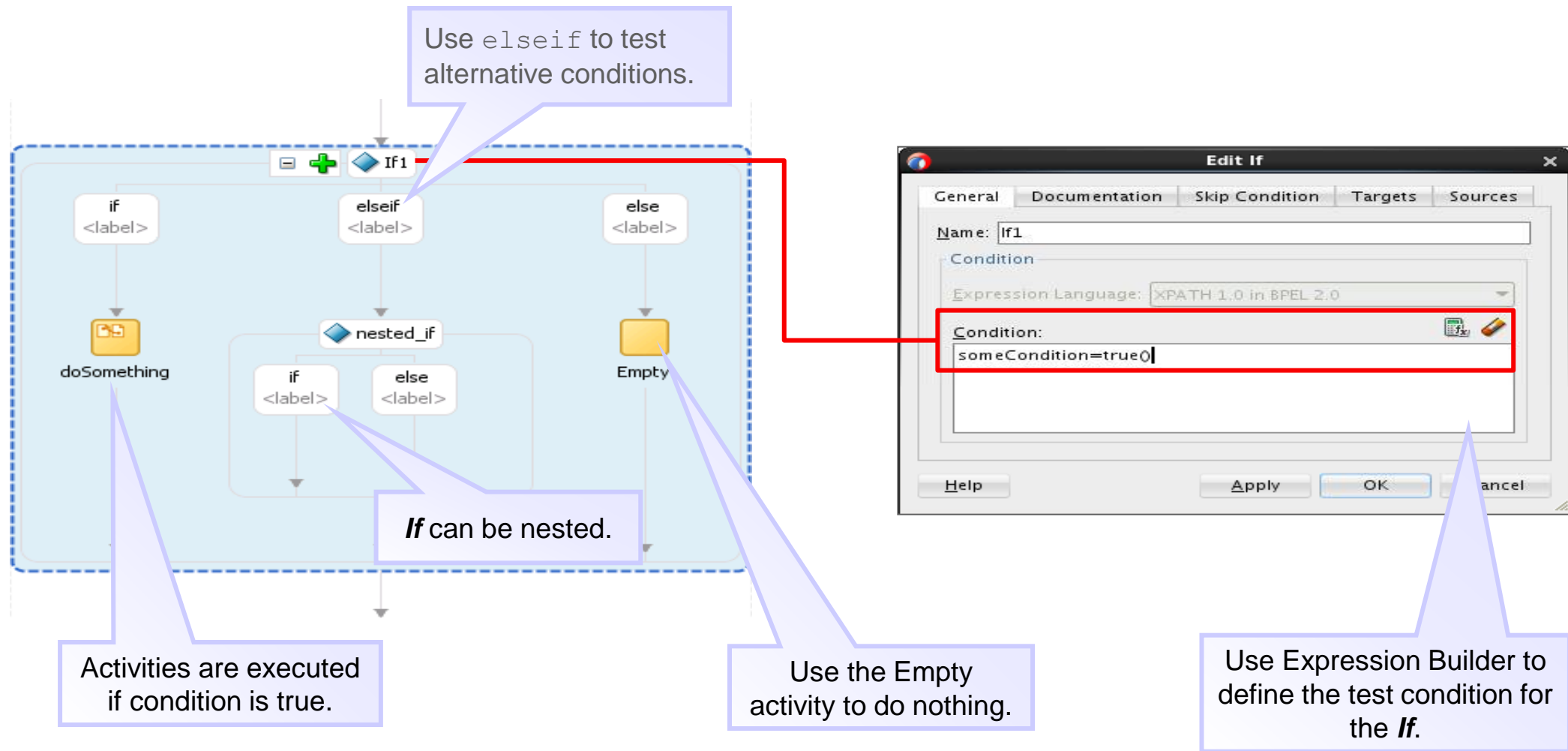


Agenda

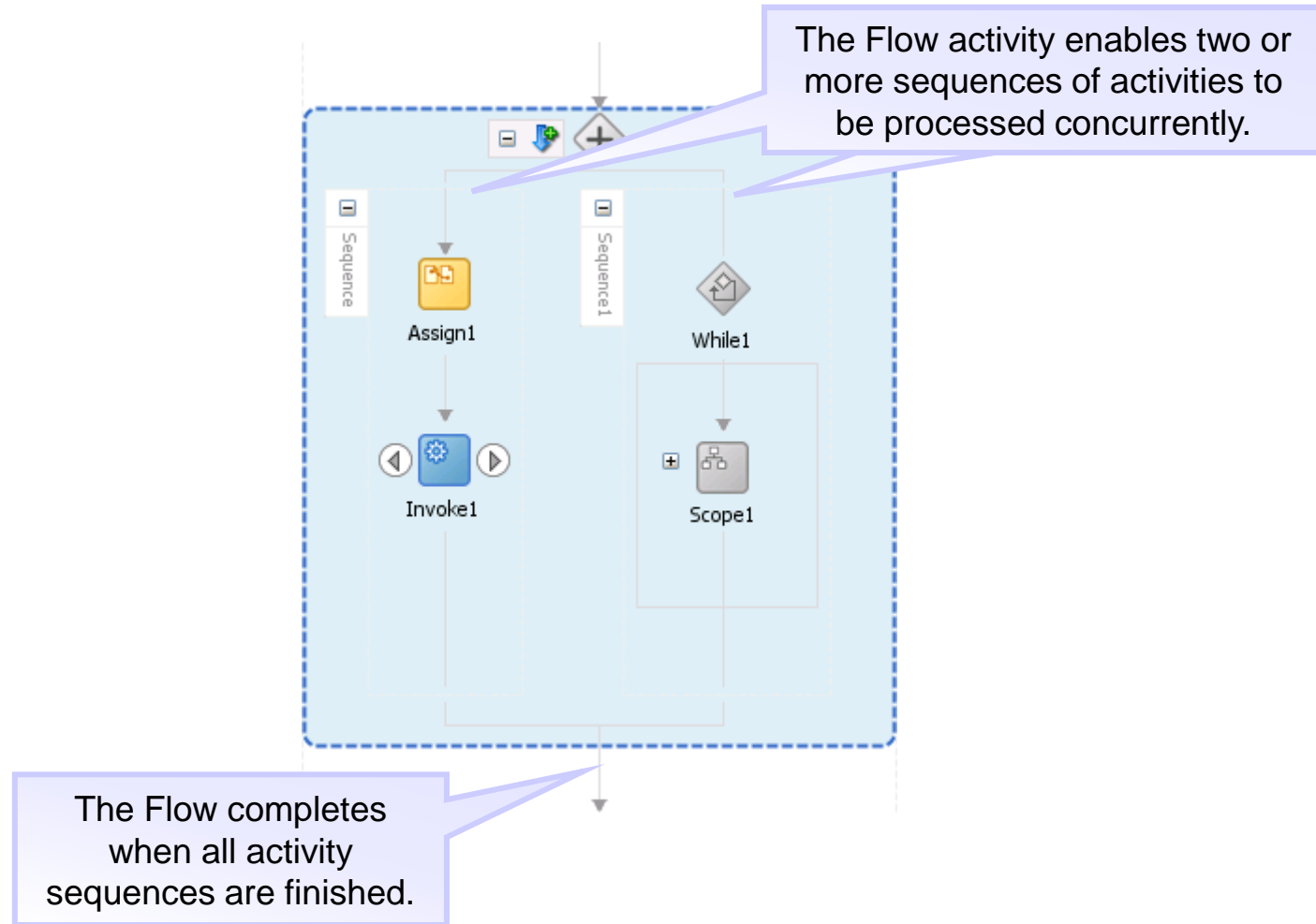
- More Activity Types
- Interaction Patterns



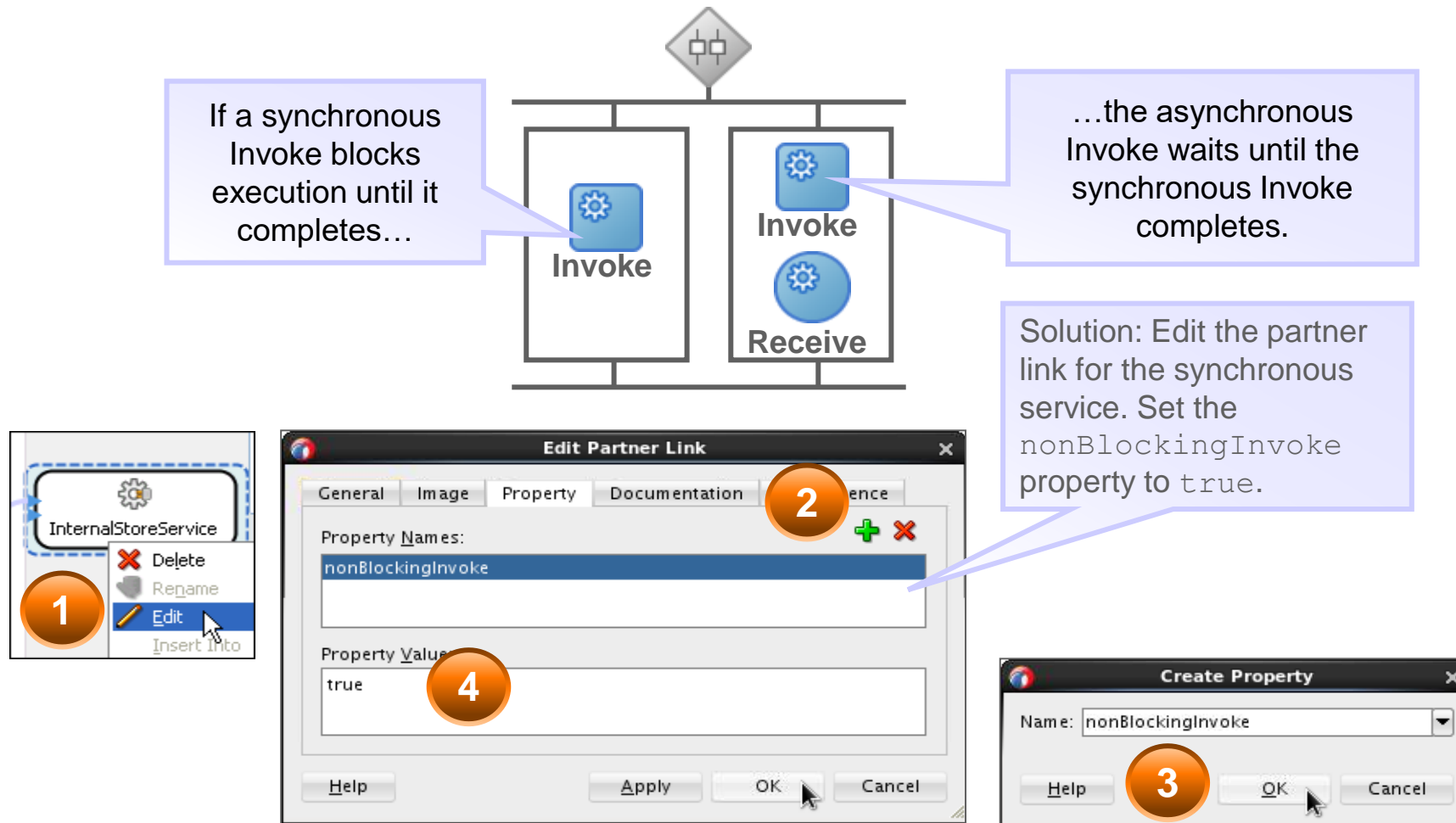
Conditional Branching with the If Activity



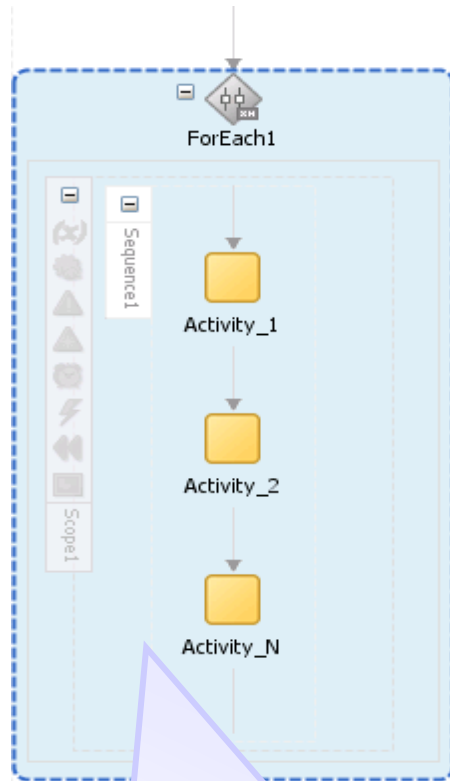
Processing with the Flow Activity



Using the nonBlockingInvoke Property



Parallel and Sequential Looping with the forEach Activity



In BPEL 2.0, the `forEach` activity performs looping iterations sequentially *N* times over a given set of activities.

The 'Edit For Each' dialog box is shown with the 'General' tab selected. The 'Name' field is set to 'ForEach_Item' and the 'Counter Name' field is set to 'ForEach1Counter'. The 'Parallel Execution' checkbox is checked.

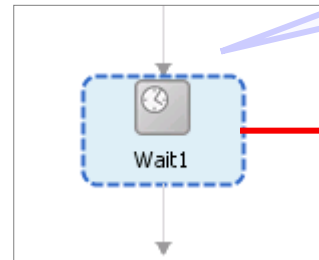
Both *sequential* and *parallel* looping are supported.

The 'Edit For Each' dialog box is shown with the 'Counter Values' tab selected. The 'Start Value' section has an 'Expression Language' dropdown set to 'XPath 1.0 in BPEL 2.0' and an 'Expression' field containing '1'. The 'Final Value' section also has an 'Expression Language' dropdown set to 'XPath 1.0 in BPEL 2.0' and an 'Expression' field containing '\$numItems'.

Declare the starting and final values for the loop counter.

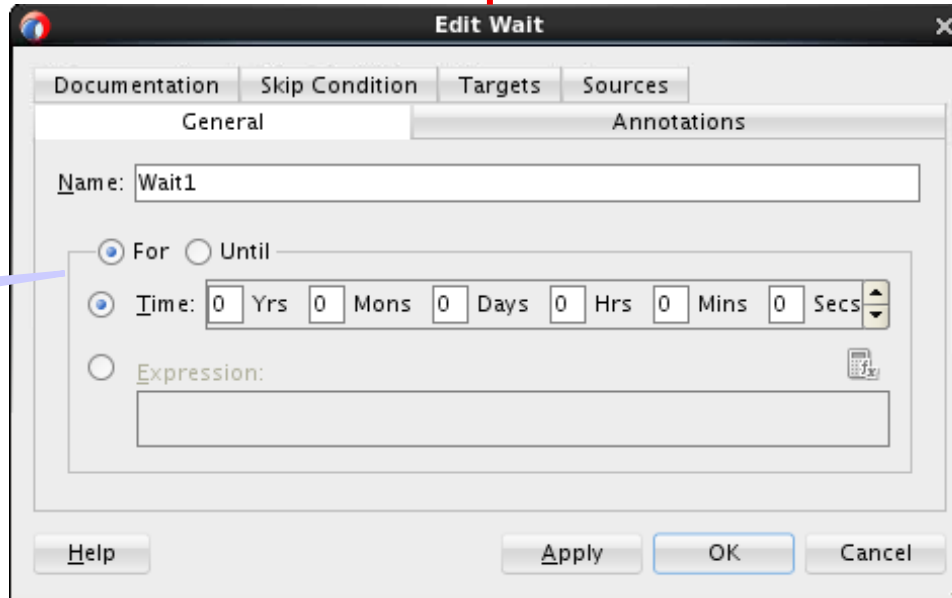
Suspending a Process with the Wait Activity

You can suspend a BPEL process by using the Wait activity.



The diagram shows a BPEL process flow. A vertical line with a downward arrow enters a rounded rectangle labeled 'Wait1'. Inside 'Wait1' is a clock icon. A red line connects the 'Wait1' activity to the 'Edit Wait' dialog box.

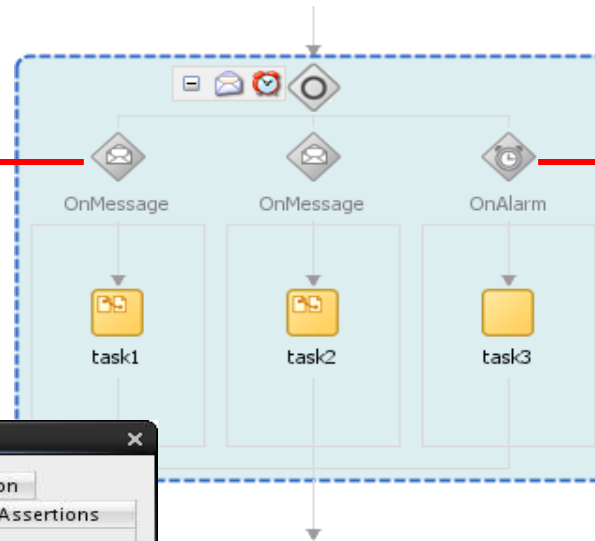
The Wait activity pauses for a specified amount of elapsed time or until a specified date-time.



The 'Edit Wait' dialog box is shown with the 'General' tab selected. The 'Name' field is 'Wait1'. The 'For' radio button is selected, and the 'Time' field is set to 0 Yrs, 0 Mons, 0 Days, 0 Hrs, 0 Mins, and 0 Secs. The 'Expression' field is empty. The 'Annotations' tab is also visible.

Waiting for a Message with the Pick Activity

The Pick activity is used to wait for a message, just as a Receive activity does.



An `onAlarm` branch implements a wait time, and provides an optional timeout.

The 'Edit OnMessage' dialog box has tabs for Documentation, Properties, Headers, Skip Condition, General, Correlations, Annotations, and Assertions. The 'General' tab is active. It contains fields for Name, Conversation ID, and Interaction Type (set to Partner Link). Below these are fields for Partner Link, Port Type, and Operation. At the bottom, there are radio buttons for Arguments Mapping and Variable, with a Variable field.

The 'Edit OnAlarm' dialog box has tabs for General, Annotations, Documentation, and Skip Condition. The 'General' tab is active. It contains a Name field and radio buttons for For and Until. Below these are radio buttons for Time and Expression. The Time field has a time picker with values for Yrs, Mons, Days, Hrs, Mins, and Secs. The Expression field has a text input and a help icon.

Looping with the While Activity

The image illustrates the configuration of a While activity. On the left, a BPMN diagram shows a 'While1' activity (diamond icon) containing a single 'Activity' (rounded rectangle icon). A callout points to the 'Activity' with the text: 'The While contains a single activity, which may be a Sequence or a Scope.'

On the right, the 'Edit While' dialog box is shown. It has tabs for 'Documentation', 'Skip Condition', 'Targets', 'Sources', 'General', and 'Annotations'. The 'General' tab is active, showing the 'Name' field set to 'While1'. Under the 'Condition' section, the 'Expression Language' is set to 'XPath 1.0 in BPEL 2.0'. The 'Condition' field is empty, with a callout pointing to it stating: 'The condition is defined with Expression Builder.'

Inside the 'Condition' field, there is a callout that says: 'The While executes its activity in a loop if its condition is true.'

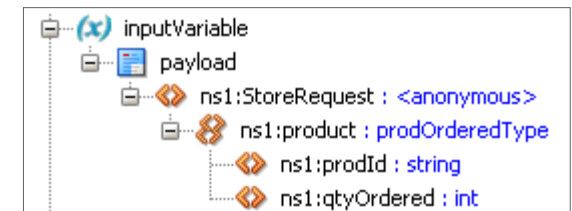
At the bottom of the dialog are buttons for 'Help', 'Apply', 'OK', and 'Cancel'.

Indexing XML Arrays Dynamically

```
<while name="While1">
  <condition>$i <= $numItems and $inStock=true()</condition>
  <sequence name="Sequence1">
    <assign name="Assign_prodid">
      <copy>
        <from>$inputVariable.payload/ns2:product[position()=$i]/ns2:prodId</from>
        <to>$Invoke_check_getInternalStoreView1_InputVariable.parameters/ns4:prodId</to>
      </copy>
    </assign>
  </sequence>
</while>
```

To access the n th element of an XML array, use the XPath expression syntax:
/parent/array[n]/child

For dynamic index expression, use either the XPath syntax [position()=num] or simply the value [num].



Quiz



The Flow and forEach activities are the two looping constructs that are available in BPEL.

- a. True
- b. False



Basic BPEL 2.0 Activities

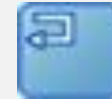
Web Services



Receive



Invoke

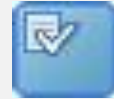


Reply

Updating Variables



Assign



Validate



Transform



XQTransform

Fault and Error Handling



Throw



Rethrow



Catch



Catch All



Compensate



CompensateScope

Miscellaneous



Wait

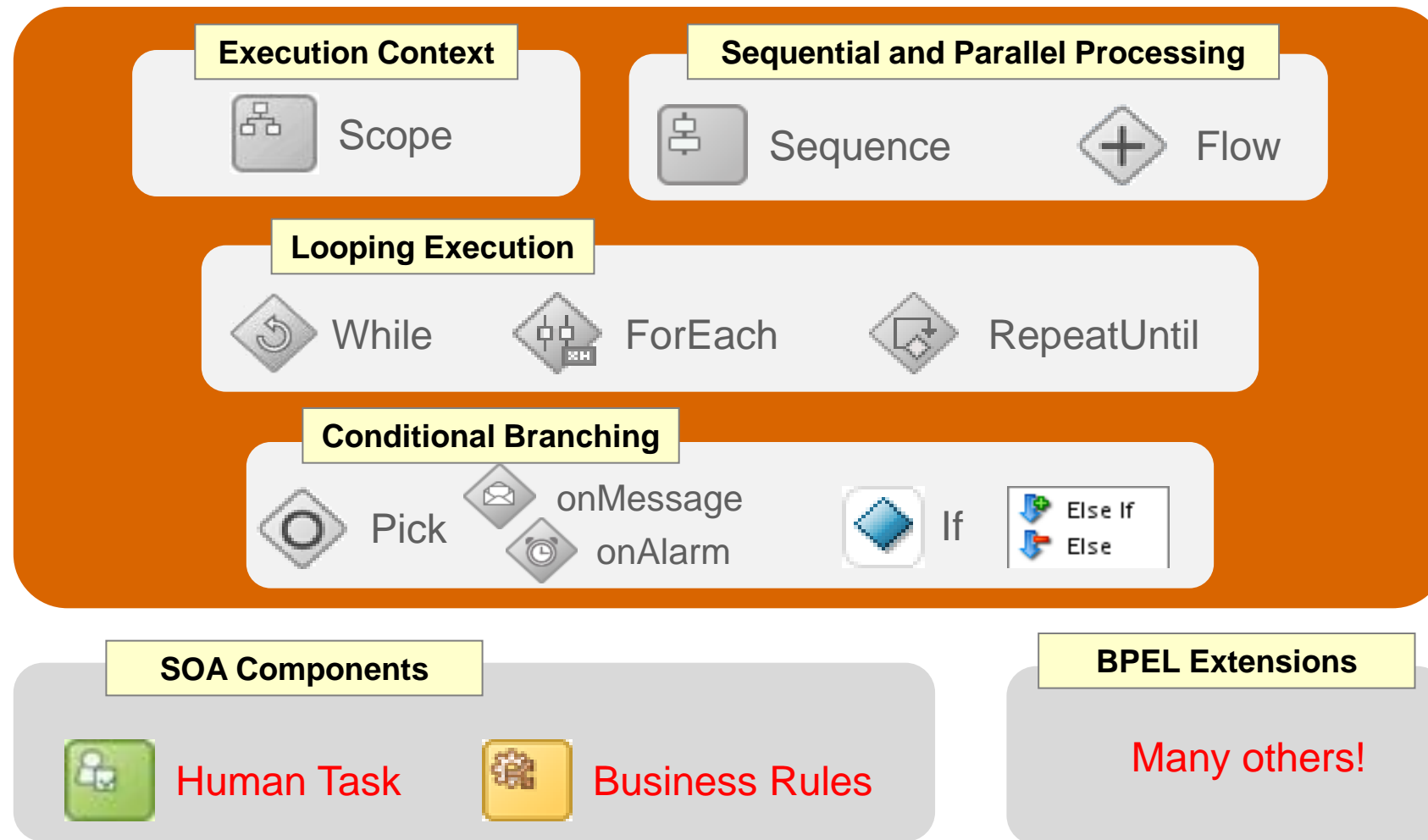


Exit



Empty

Structured and Extension BPEL 2.0 Activities



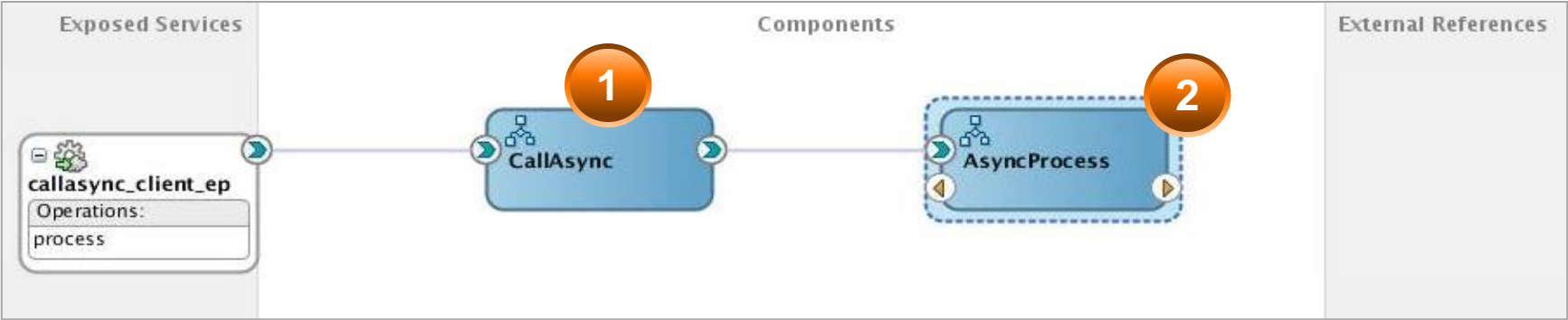
Summary

In this lesson, you should have learned how to:

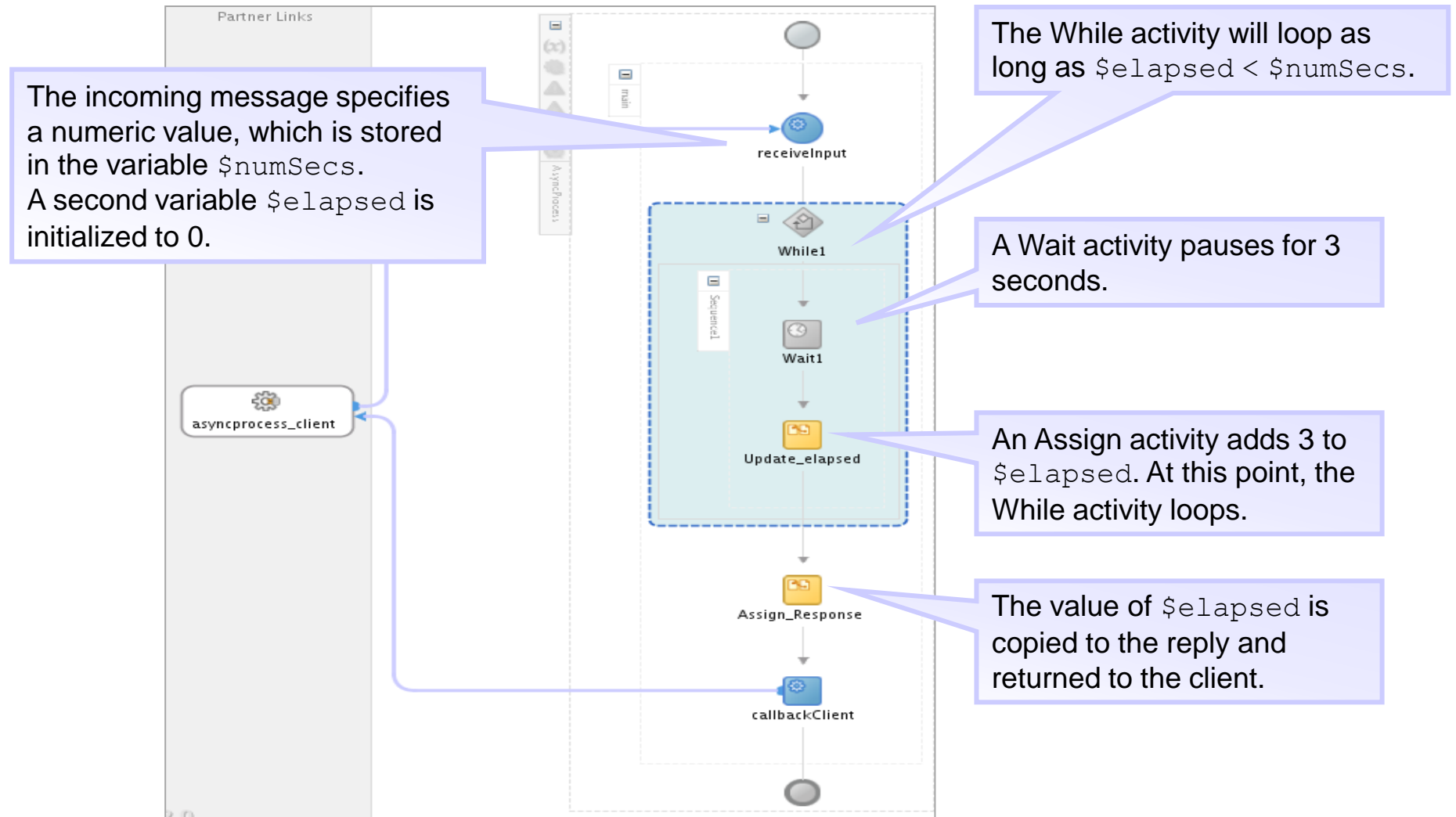
- Perform conditional branching by using an If activity
- Implement parallel processing by using a Flow activity
- Implement non-blocking invocation with a Flow
- Create parallel branches dynamically with a forEach activity
- Implement a Pick activity with an alarm and a timeout
- Execute activities repetitively with a While activity
- Explore various SOA component interaction patterns
- Explore a dynamic service invocation pattern



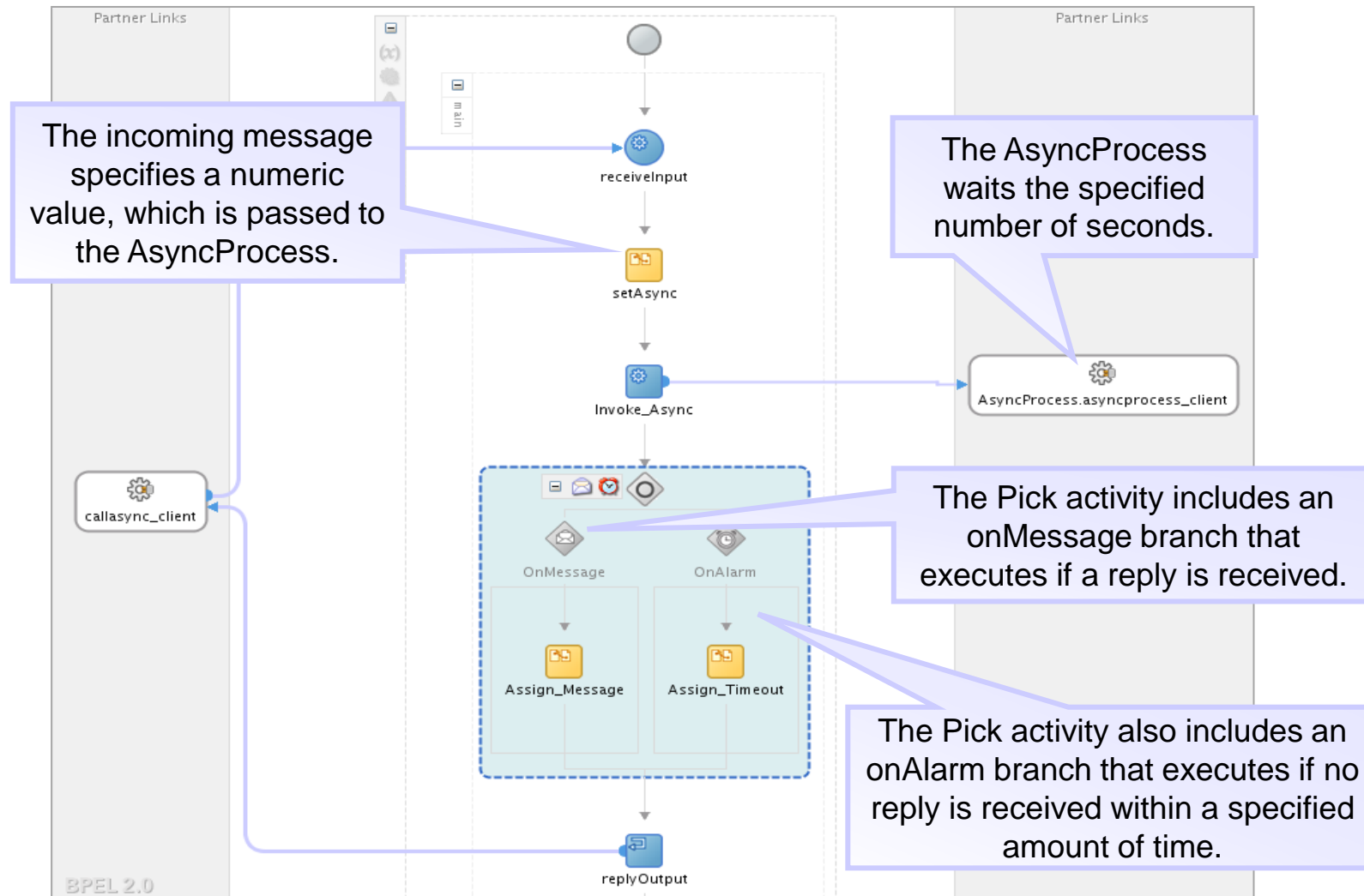
Practice 7 Overview



Practice 7 Overview



Practice 7 Overview



Before You Begin

The instructions for this practice and those that follow include path references that include XML namespace prefixes, for example:

- “Use Expression Builder to define the expression
`string($inputVariable.payload/ns4:shipMethod).`”

If you have built each of the components in the order that was prescribed in the instructions, Expression Builder generates an expression that includes namespaces that match.

If you have built components in a different order, or deleted, and then re-created components, *your namespace prefixes may not match what is listed in these instructions.*

- In that case, *use the namespace references in your project.*
- Do **NOT** change those references to match the book.