

Sharing Functionality in Oracle SOA Suite

Objectives

After completing this lesson, you should be able to:

- Use the design-time Metadata Services (MDS) Repository to share files
- Create and use component and project templates
- Create and use inline and stand-alone BPEL subprocesses
- List the differences between templates and subprocesses

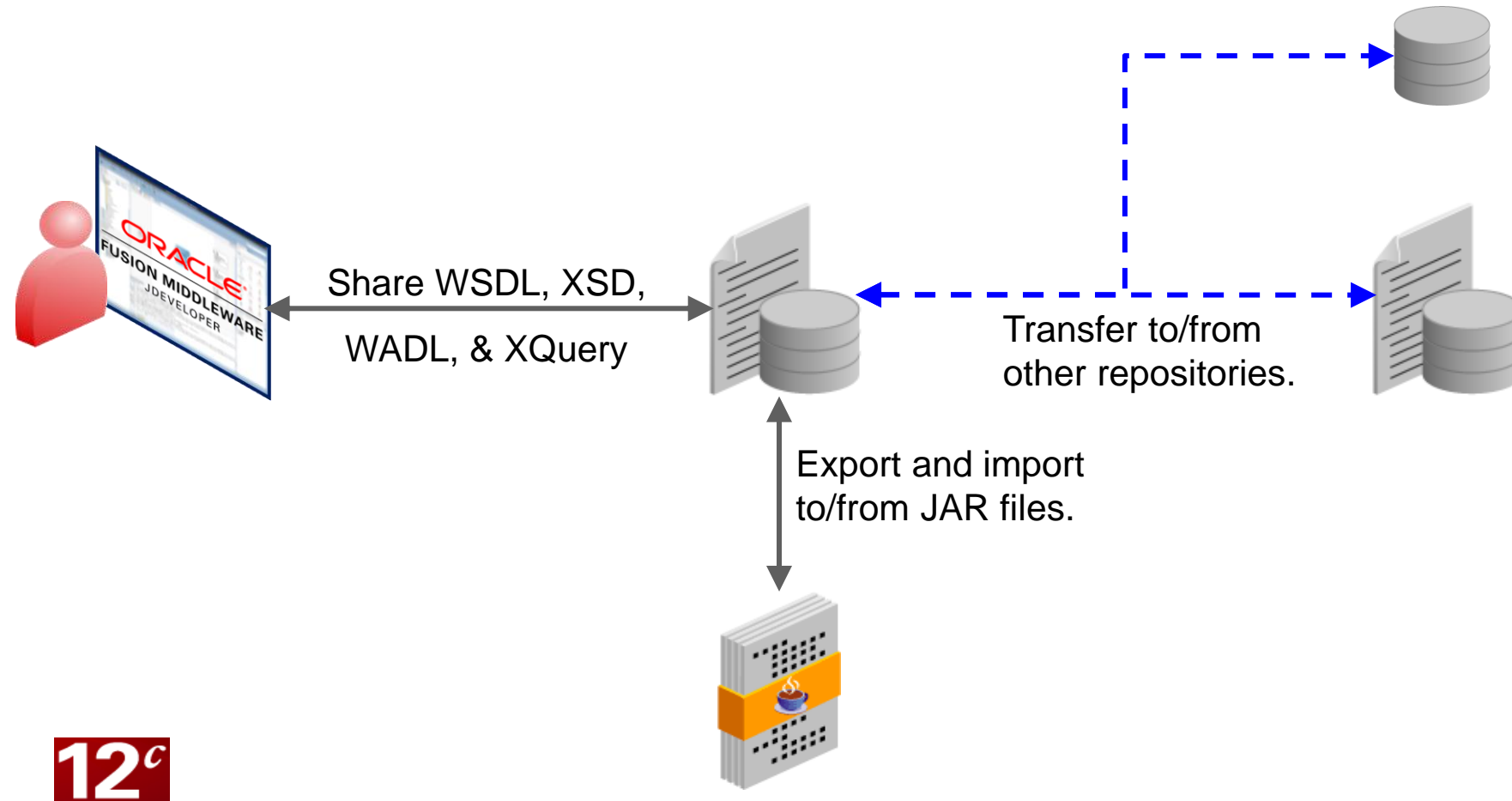


Agenda

- Design-Time Metadata Services (MDS) Repository
- Templates
- BPEL Subprocesses



Design-Time Metadata Services (MDS) Repository



Quiz



When artifacts are moved into the repository, it is important to manually include all dependencies and references.

- a. True
- b. False

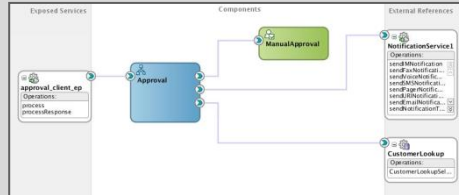


Agenda

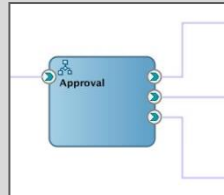
- Design-Time Metadata Services (MDS) Repository
- **Templates**
- BPEL Subprocesses



Templates: Overview



A complete **SOA project** packaged and used to start new projects

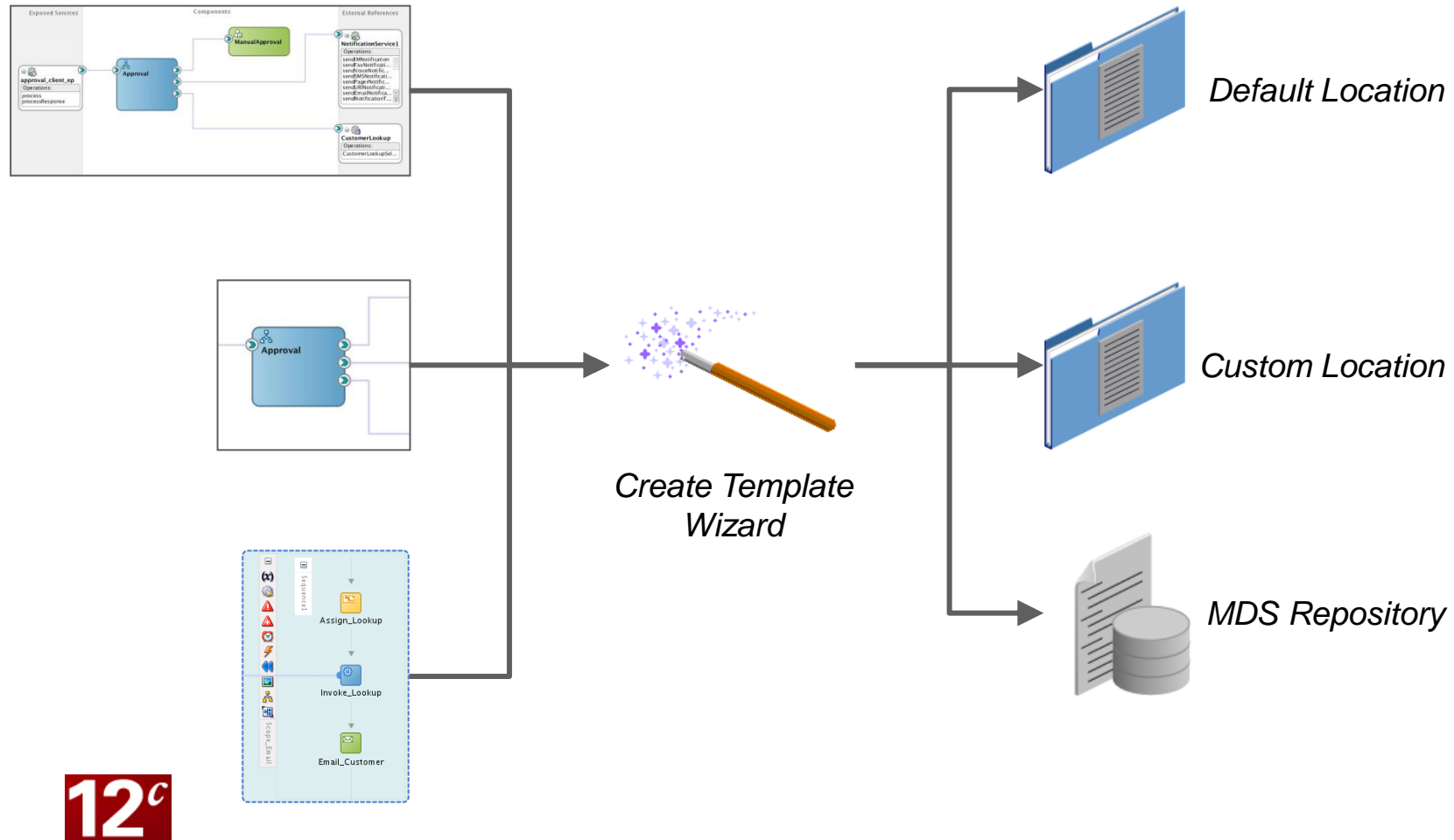


A **Service component**, such as a BPEL process, packaged for import into other projects. All dependent components and wires are also packaged.



A **Scope activity** of a BPEL process that is packaged as a custom activity in the Components window and that is ready for import into other BPEL projects

Creating and Using Templates



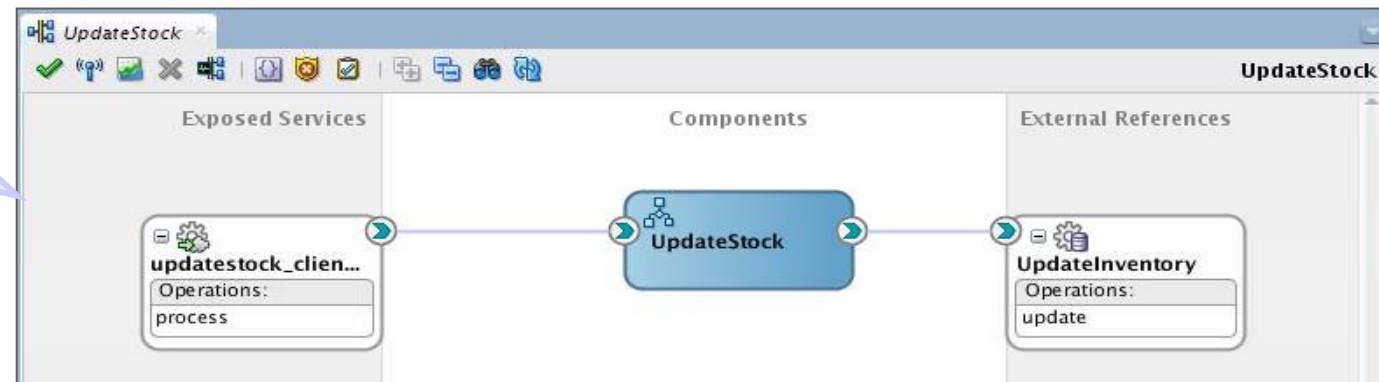
Using Project Templates



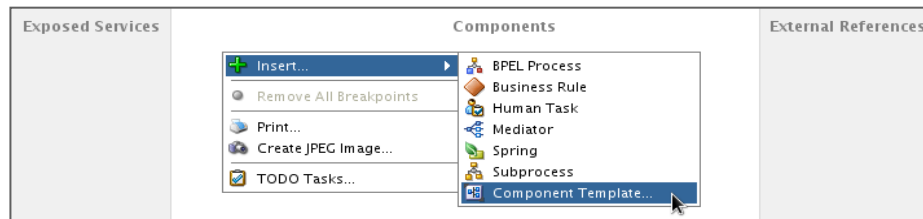
Select
Start from: SOA Template

Specify the desired
template.

The project is
created from
the template.

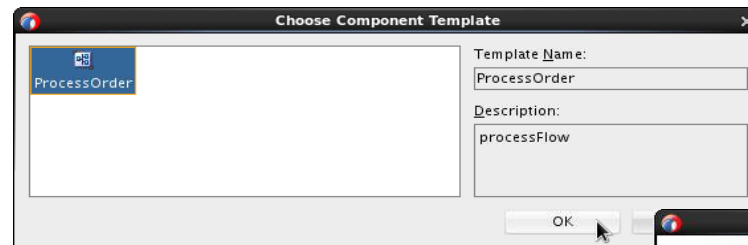


Using Service Component Templates



1

Insert a Component Template.

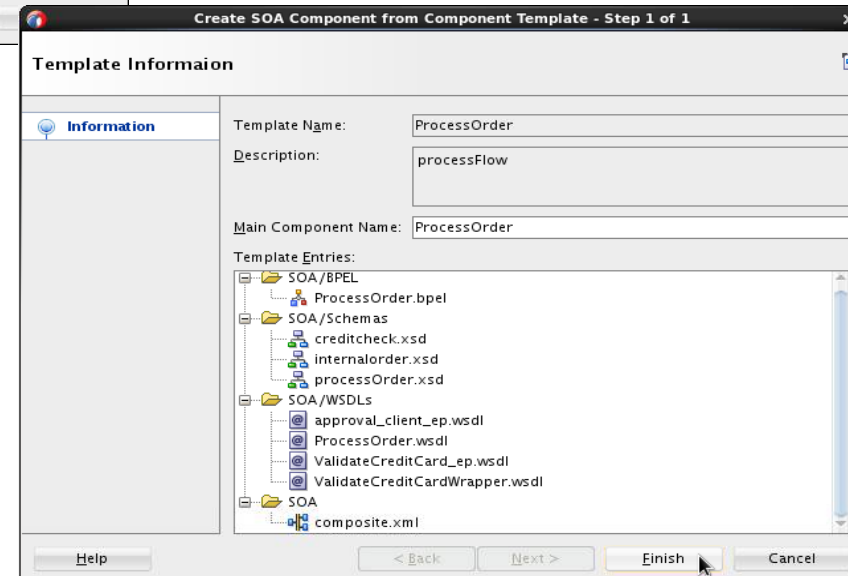


2

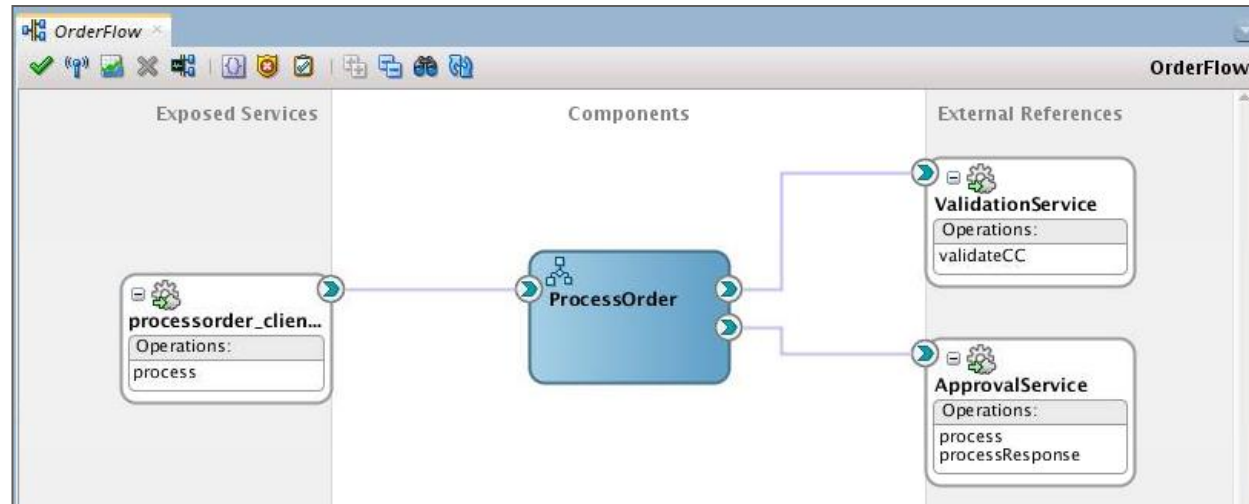
Select the specific template.

3

Review the template information and click Finish.



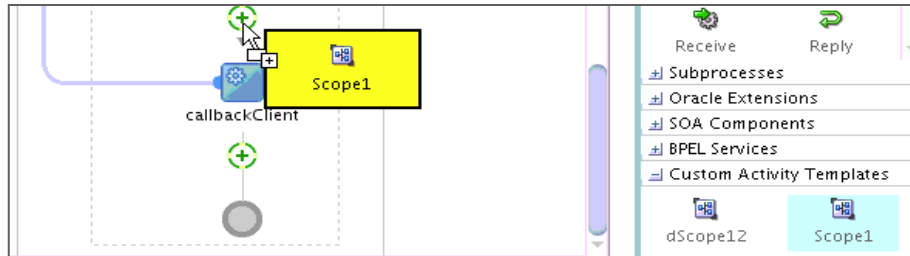
Using Service Component Templates



4

The component and any dependencies are added to the project.

Using BPEL Scope Activity Templates

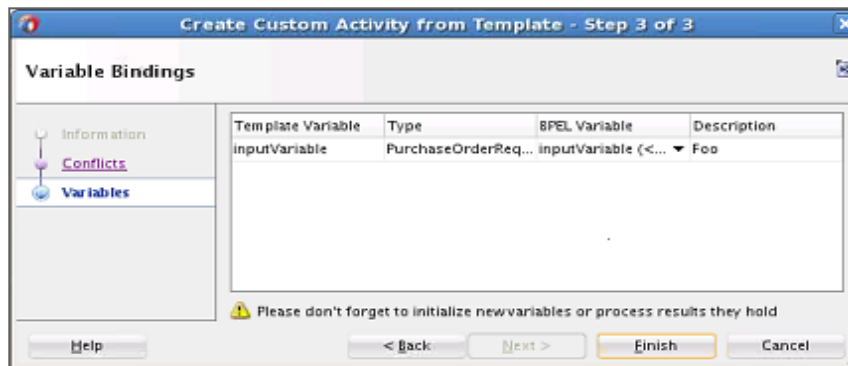
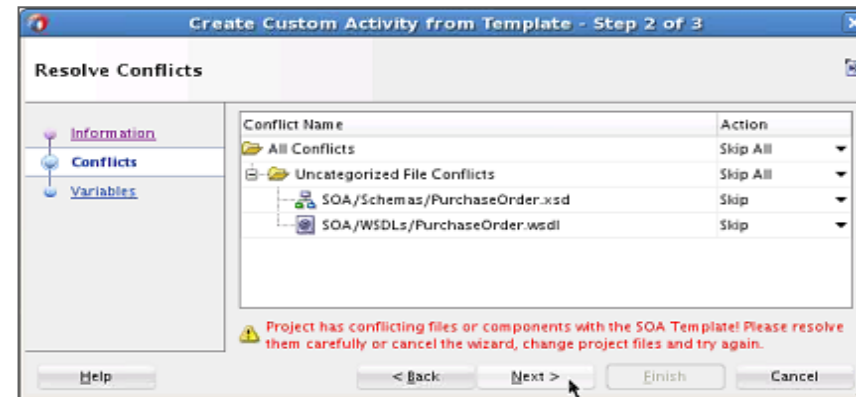


Add the activity from the Component Palette.

1

Resolve any conflicts (if the project references any files that are also named in the template).

2



Bind any variables that are referenced in the template to those found in the BPEL project.

3

Quiz



Which of the following are types of templates?

- a. SOA Project
- b. Partner Link
- c. Component
- d. External Reference
- e. Scope Activity



Agenda

- Design-Time Metadata Services (MDS) Repository
- Templates
- BPEL Subprocesses



BPEL Subprocesses: Introduction

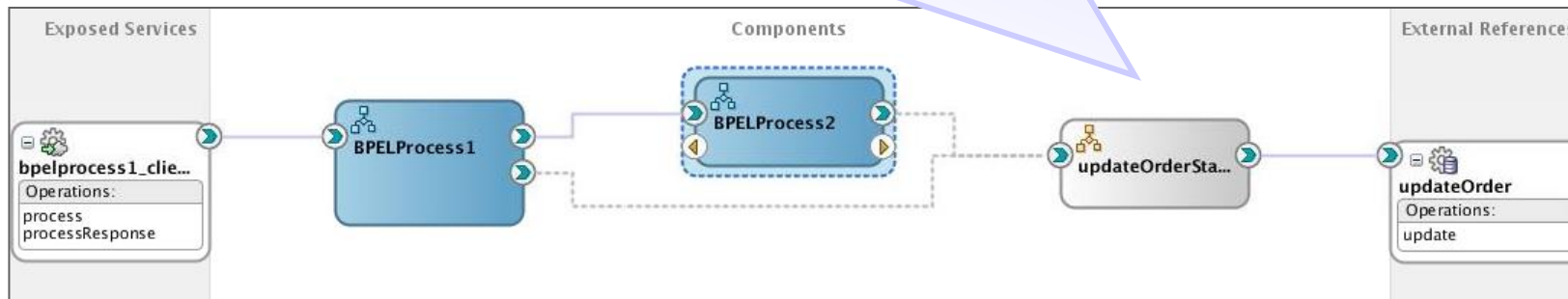
- Are process fragments that modularize a reusable piece of business logic
- Can include partner links and references
- Can be invoked from other “parent” BPEL processes through a *call* activity
- Do not have an interface (compares to a subroutine in an object-oriented programming language)
- Execute completely in the context of the parent process, permitting access to process variables and inheriting fault handling and compensation logic

12^c

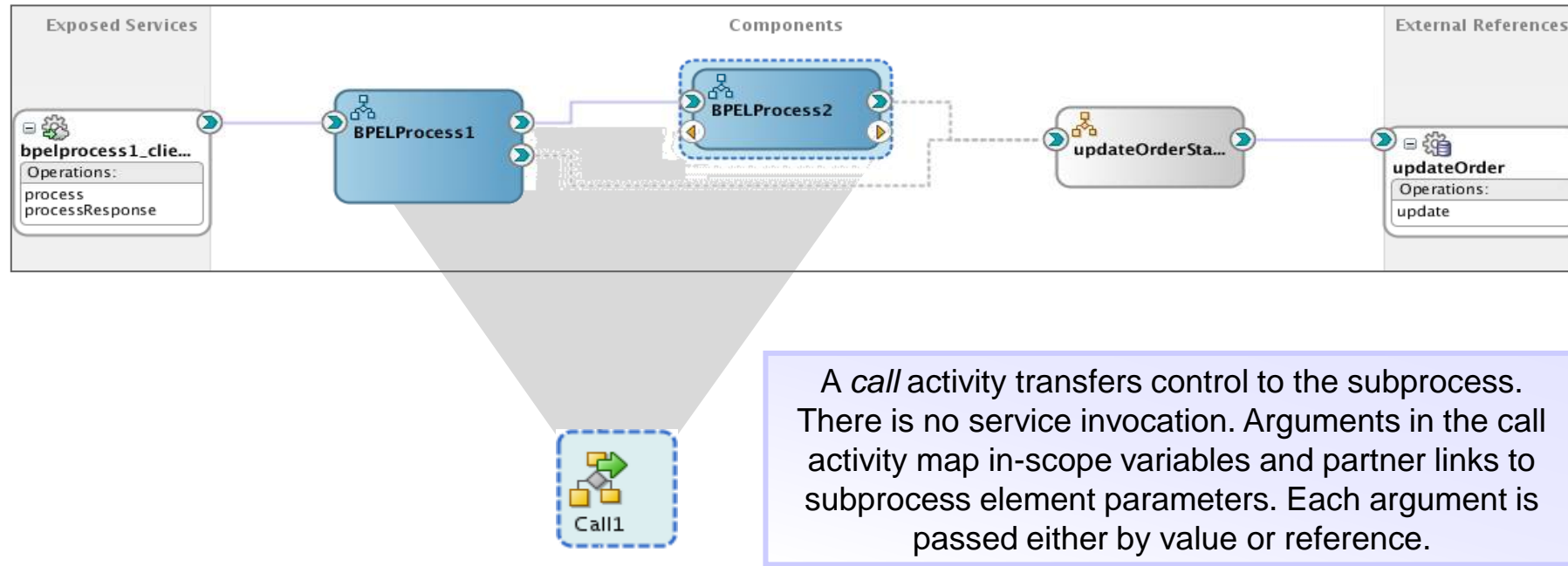
ORACLE®

Stand-Alone Subprocess

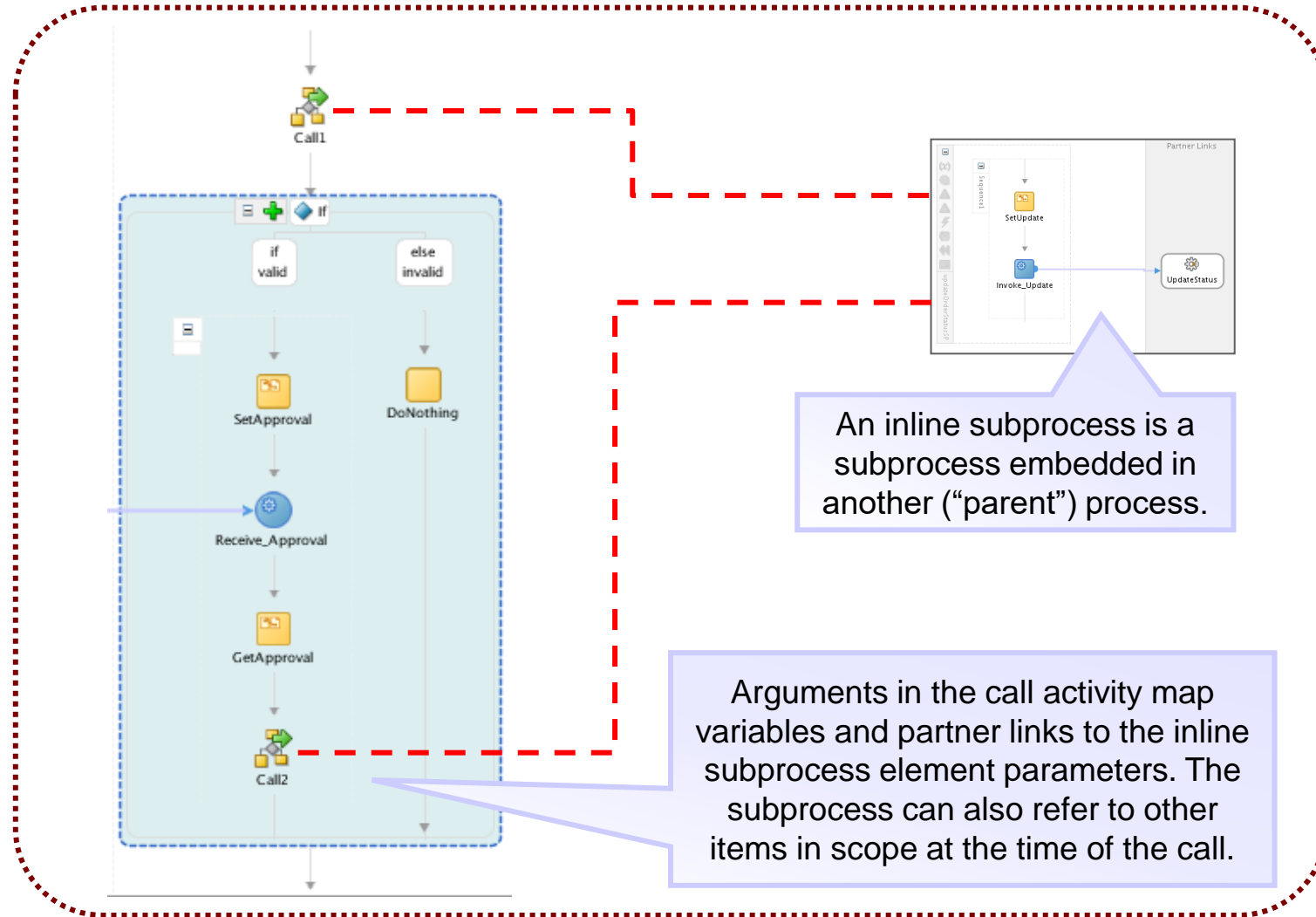
A stand-alone subprocess is a self-contained snippet of process logic. All references resolve to local definitions or arguments that are defined in the partner links and variables.



Calling a Stand-Alone Subprocess



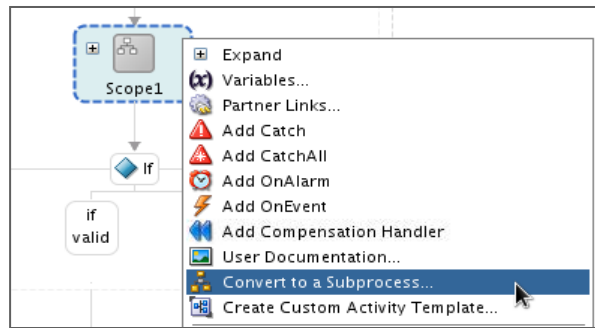
Inline Subprocess



Comparing Stand-Alone and Inline Subprocesses

Stand-Alone Subprocess	Inline Subprocess
<ul style="list-style-type: none">• They are available in BPEL 2.0 only.• A BPEL <i>call</i> activity invokes the subprocess.• They are visible in the Components window.	
Is a fragment of a BPEL process that includes a number of activities that are reused across other BPEL processes	Is useful for groups of activities that are reused within one BPEL process
Can be called from any BPEL process in the same composite	Is part of the parent BPEL process code and is not visible in the composite view
Does not have an interface, and can be called only from another BPEL process. It can include partner links.	Can be called only from the parent BPEL process. It can use the partner links and in-scope variables of the parent process.
Defines the parameters to set	Can either define the parameters to set or can use parent process values

Creating an Inline Subprocess

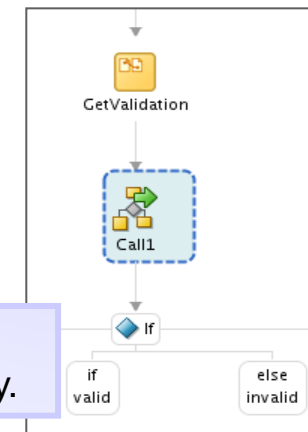


1 Right-click the scope activity and select *Convert to a Subprocess*.

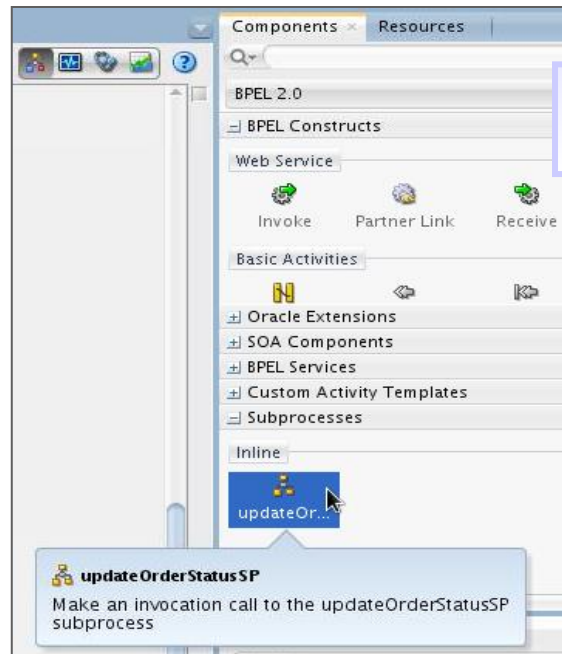
The dialog box is titled 'Create Inline Subprocess'. It contains the following fields and options:
- Name: updateOrderStatusSP
- ☒ Replace Scope with Subprocess Call
- Label: (empty)
- Comment: (empty)
- Image: (empty)
Buttons: OK, Cancel

2 Assign a name, and if desired, add a label and/or an image.

3 By default, the scope is converted to a call activity.

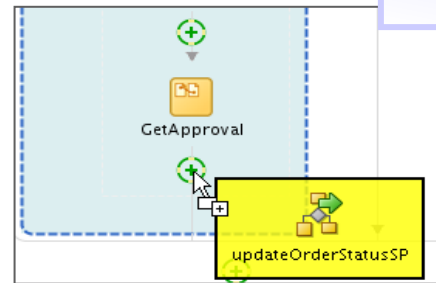


Calling an Inline Subprocess



The subprocess is available from the Component Palette.

1

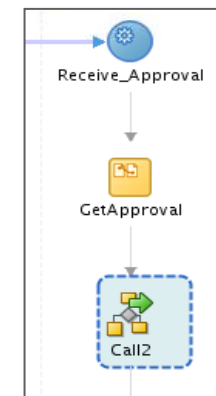


Drag and drop the subprocess onto the BPEL process.

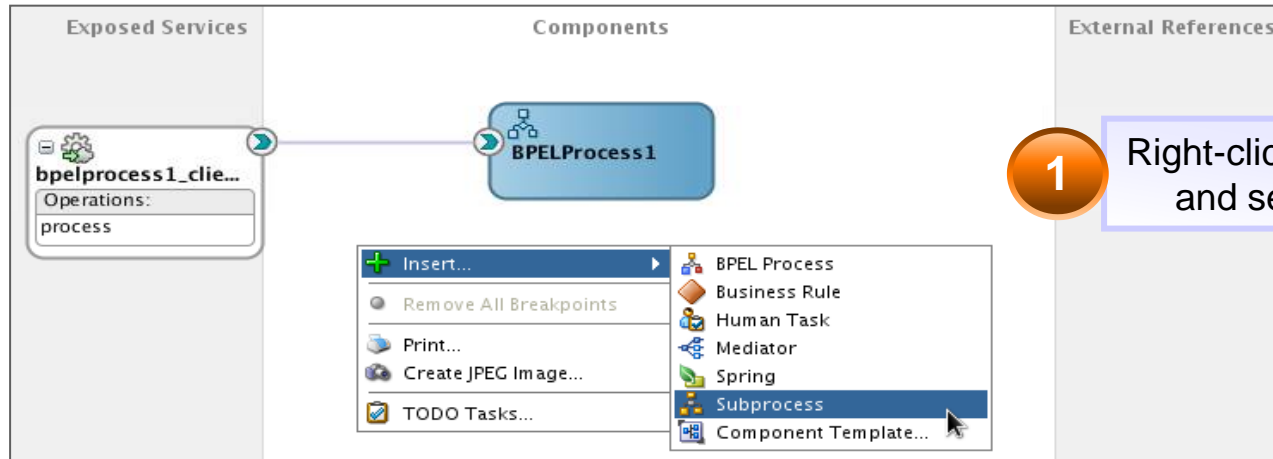
2

3

A call activity is generated.



Creating a Stand-Alone Subprocess

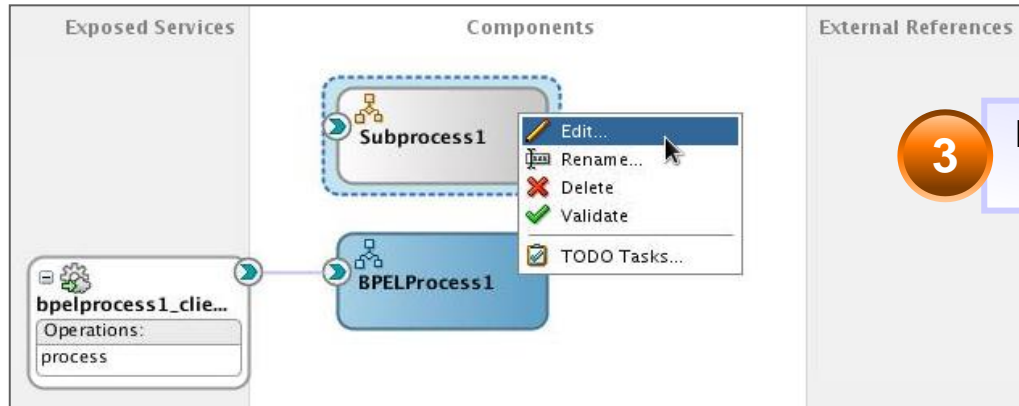


1 Right-click the Components swimlane and select Insert > Subprocess.

2 Name the subprocess and click OK.

The 'Create Subprocess' dialog box is shown. It has a title bar 'Create Subprocess' and a close button. The main area is titled 'Subprocess' and contains a description: 'A subprocess is a self contained reusable BPEL module. It can be invoked by other BPEL processes in order to have a smaller footprint and allow sharing of BPEL logic between multiple processes and composites.' Below the description are three input fields: 'Name:' with the value 'Subprocess1', 'Namespace:' with the value 'http://xmlns.oracle.com/Samples/StandAlone/Subprocess1', and 'Directory:' with the value 'u01/app/fmw12c/domains/mywork/Samples/StandAlone/SOA/BPEL'. At the bottom are buttons for 'Help', 'OK', and 'Cancel'.

Creating a Stand-Alone Subprocess

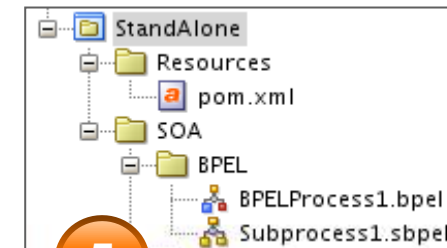
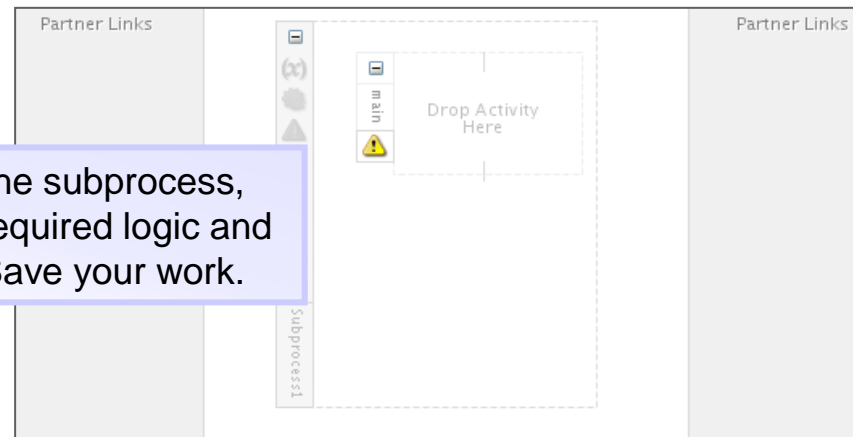


3

Right-click the subprocess and select Edit.

4

Configure the subprocess, adding the required logic and variables. Save your work.

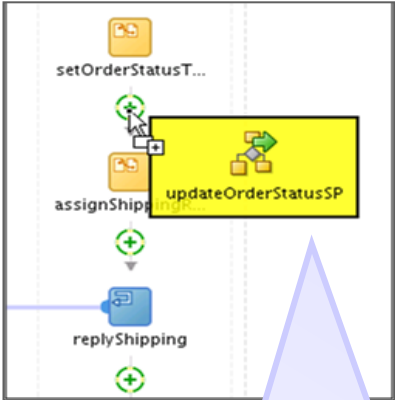


5

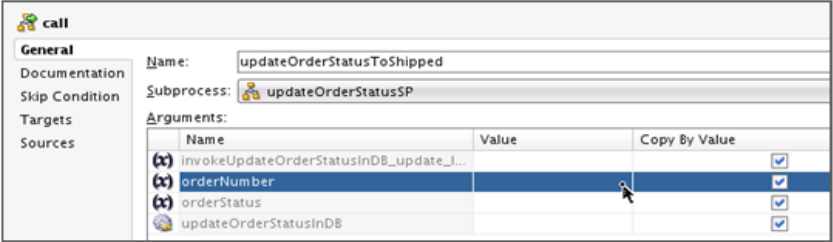
The subprocess is stored in the project as an *.sbpel* file.

Calling a Stand-Alone Subprocess from BPEL


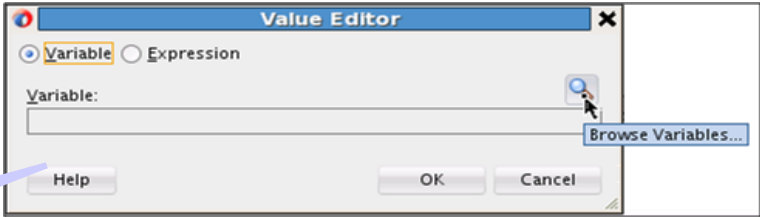
1 Add the activity from the Component Palette.



2 Map parent process values to subprocess arguments.



Name	Value	Copy By Value
invokeUpdateOrderStatusInDB_update_I...		<input checked="" type="checkbox"/>
orderNumber		<input checked="" type="checkbox"/>
orderStatus		<input checked="" type="checkbox"/>
updateOrderStatusInDB		<input checked="" type="checkbox"/>



Differences Between Templates and Subprocesses

A **template** is a customizable, skeletal composite, process, or scope activity. You can drag and drop a template onto a SOA composite application or a BPEL process and make additional changes. You essentially are copying and pasting a template. For example, if there are 50 lines of code in a template and you copy it twice to use, the code increases by 100 lines.

A **subprocess** is a BPEL code snippet that is intended for a specific purpose. A subprocess can be called and used as it is. An inline subprocess of 50 lines can be called twice and the parent process code remains at 50 lines, and not 100. Subprocesses perform better and have a smaller memory footprint than templates.

Quiz



Inline subprocesses can make use of any partner links and/or variables that are in scope at the time of the call activity.

- a. True
- b. False



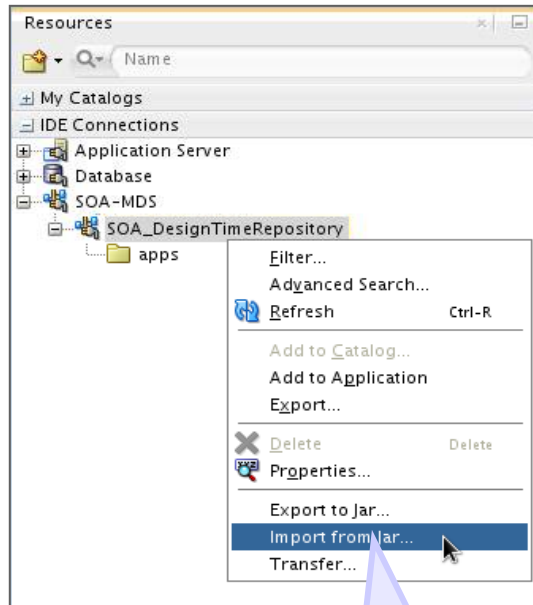
Summary

In this lesson, you should have learned how to:

- Use the design-time Metadata Services (MDS) Repository to share files
- Create and use component and project templates
- Create and use inline and stand-alone BPEL subprocesses
- List the differences between templates and subprocesses



Practice 10 Overview

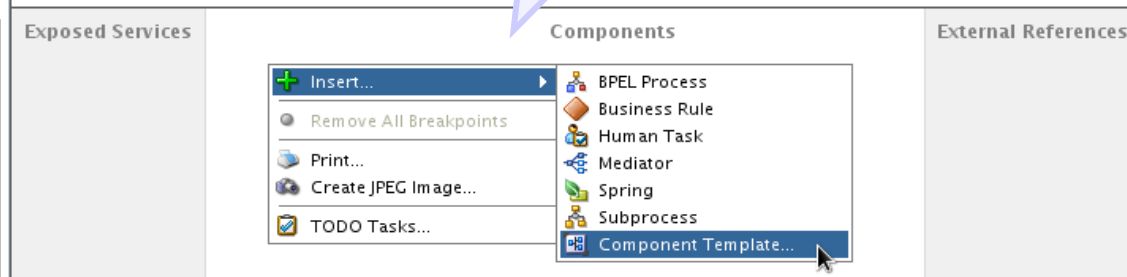


1

You first import a collection of templates from a JAR file.

2

You use a component template in the building of a project.



3

You create and call an inline BPEL subprocess.

