14

Testing and Debugging Composite Applications

Objectives

After completing this lesson, you should be able to:

- Create test suites for composite applications
- Create test cases to initiate inbound messages, and to emulate outbound, fault, and callback messages
- Create test cases with value-based and XML-based assertions
- Discuss strategies for debugging and troubleshooting applications
- Use the SOA debugger to step through an application and observe values during execution

Agenda

- Configuring Test Cases
- Running Test Cases
- Using the SOA Debugger

Testing SOA Composite Applications

The goals of testing a SOA composite application are to:

- Automate testing of interactions between internal components, as well as with external services
- Simulate interactions between a SOA composite application and its partner web services
- Ensure that a composite application interacts with web service partners as expected before deployment in a production environment

Introducing the Composite Test Framework

The composite test framework:

- Simulates web service partner interactions
- Validates process actions with test data
- Creates reports of test results
- Supports the creation of tests at the SOA composite application–level for:
  - Wires
  - Service binding components
  - Service components (such as BPEL processes and Mediator service components)
  - Reference binding components

Emulations and Assertions

The composite test framework allows you to create *emulations* and *assertions*.

- Emulations enable you to simulate the message data that your SOA composite application receives from both the service components inside the composite and the binding components outside the composite.
- Assertions enable you to validate an entire XML document, a part of a message, a non-leaf element, or a leaf element at a point during SOA composite application execution by comparing actual values to expected (asserted) values.

Test Suites: Overview

For each composite application, the composite test framework supports:

- Creating one or more test suites, each comprising a collection of test cases
- Deploying the composite application with its test suites to a runtime environment
- Running a composite application test suite, called a test run, by using Oracle Enterprise Manager Fusion Middleware Control

Note: Each test run corresponds to a single composite application instance.

Test Cases: Overview

A test case:

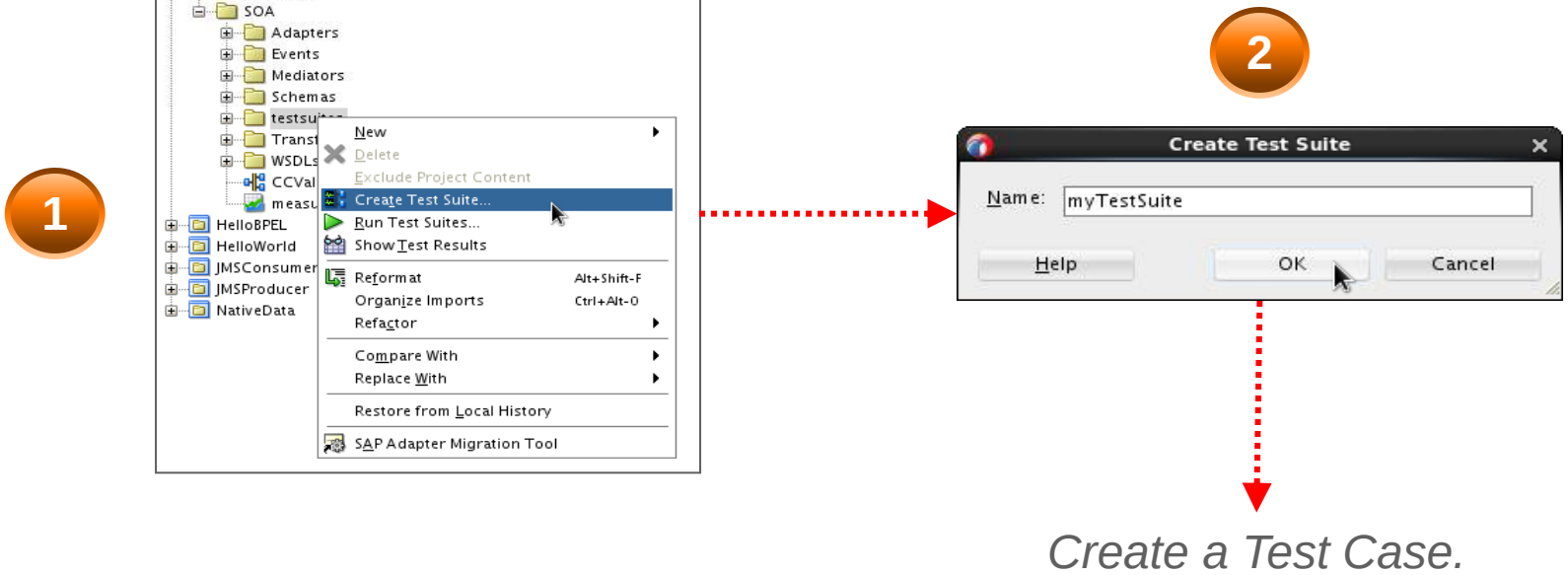
- Is a component of a test suite
- Is a specific test that is included in a test run
- Can be thought of as a unit test that comprises the following components:
  - Process initiation to initiate your process with an input payload
  - Emulations to emulate interactions
  - Assertions to validate message content for the interactions
  - Message files containing the test message samples used in the process initiation, emulations, and assertions

Contents of a Test Case

A test case can contain one or more:

- Inbound message initiations or initiating events
- Outbound message emulations
- Callback message emulations
- Fault message emulations
- Value-based or XML-based assertions

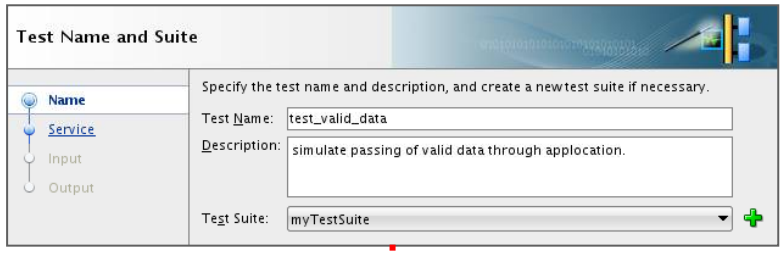
Creating a Test Suite

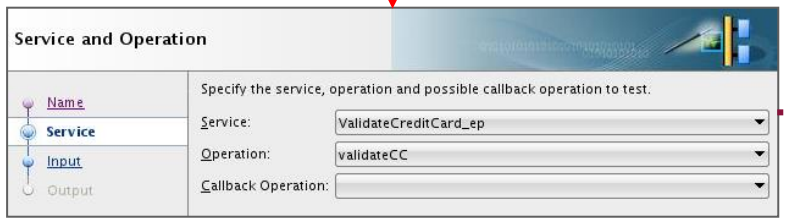


Create a Test Case.

Creating the First Test Case

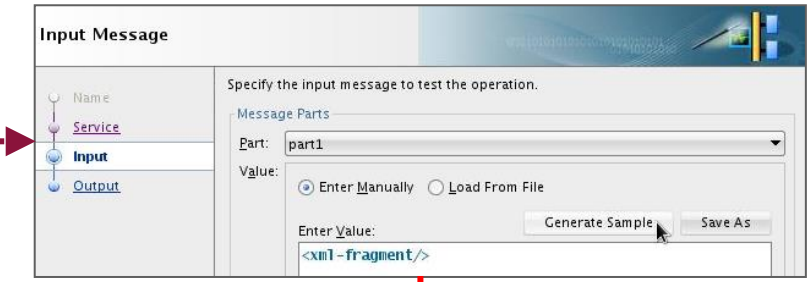
1 Name the test case.

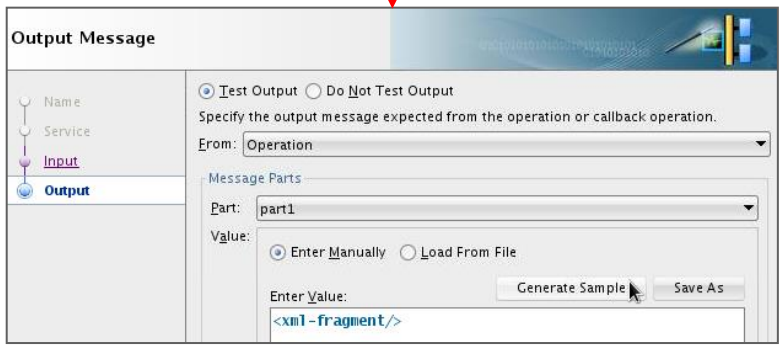




2 Specify the service, operation, and possible callback to test.

3 Load or generate the input message to be used.



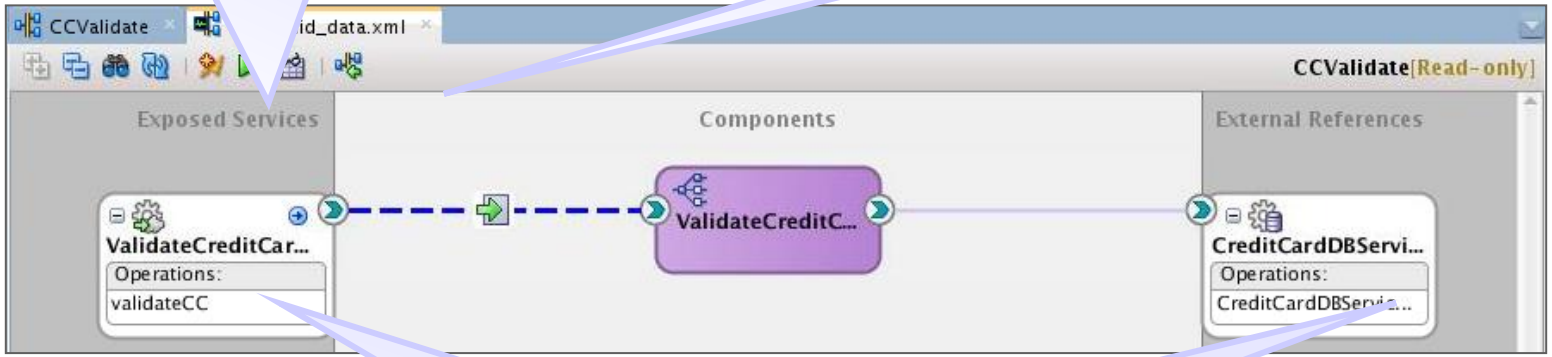


4 Load or generate the output message, or specify that output will not be tested.

Test Case Editor

A small blue arrow indicates that the initiating inbound message is defined.

The dashed wire and wire action icon indicate that an output message is defined.



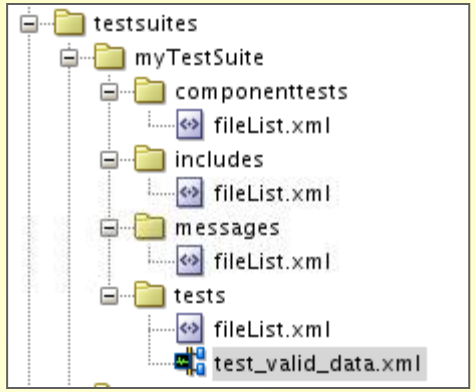
The test case editor looks like the assembly editor `(composite.xml)`, except that the outer swimlanes are differently colored.

Test Case File

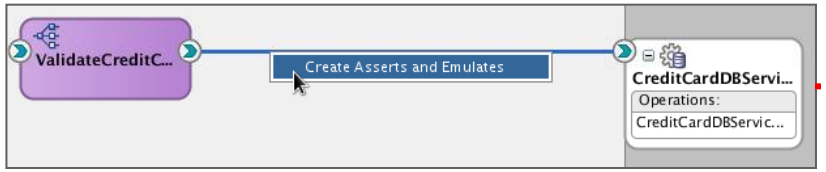
```
compositeTest compositeDN="CCValidate" xmlns="http://xmlns.oracle.com/sca/2006/test">
<initiate serviceName="ValidateCreditCard_ep" operation="validateCC" isAsync="false" delay="PT0S">
   <message>
* * *
        <CreditCheckRequest xmlns="http://www.example.org/ns/ccauthorize">
          <CCNumber>9000-1234-1234-1234</CCNumber>
          <amount>200</amount>
        </CreditCheckRequest>
* * *
   </message>
  </initiate>

  <wireActions wireSource="ValidateCreditCard_ep" operation="validateCC">
    <assert comparisonMethod="xml-similar">
      <description/>
      <expected>
        <location key="output"/>
        <message>
* * *
            <CreditCheckResponse xmlns="http://www.example.org/ns/ccauthorize">
              <status>VALID</status>
            </CreditCheckResponse>
* * *
        </message>
      </expected>
    </assert>
  </wireActions>
</compositeTest>
```

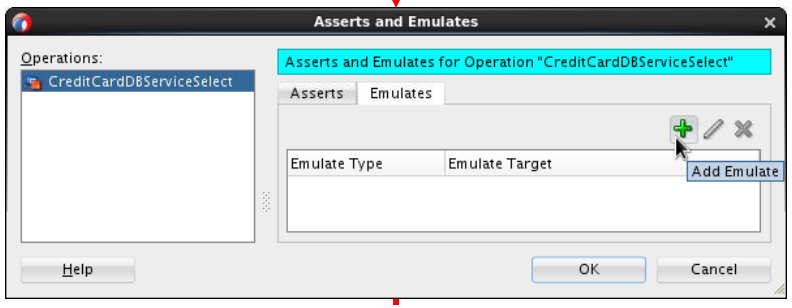
The file includes the input message.

If specified in the wizard, It also includes a wire assertion and the description of the output message.

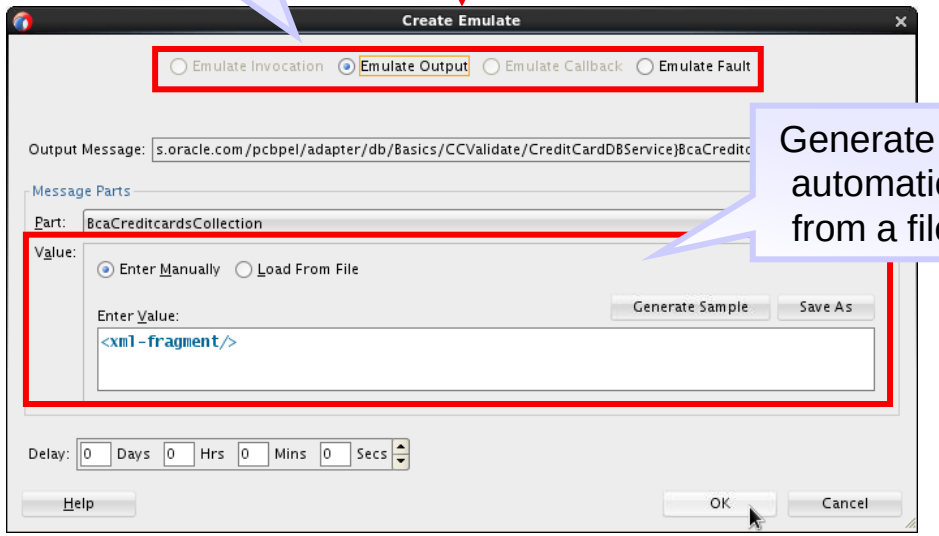


Creating Emulations





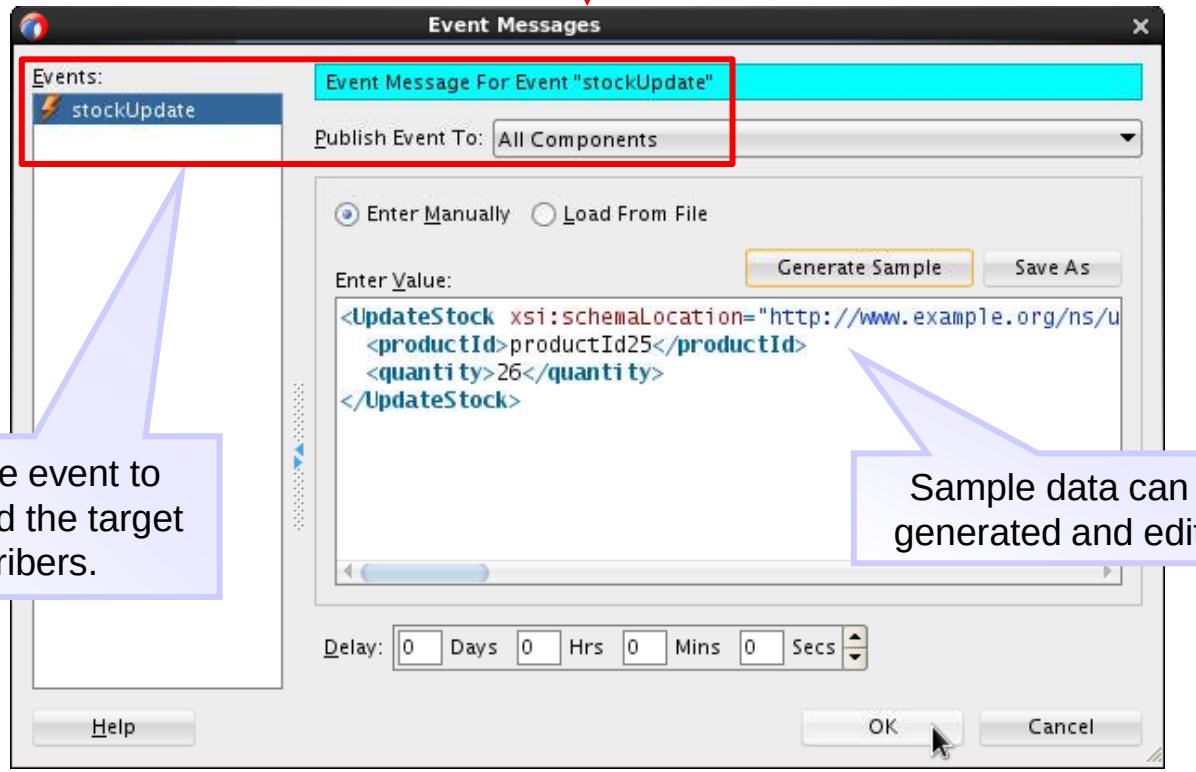
You can emulate output, callback, and fault messages, depending on the context of the selected wire.



Generate sample messages automatically, or load them from a file or manual input.

Emulating Events in a Test Case

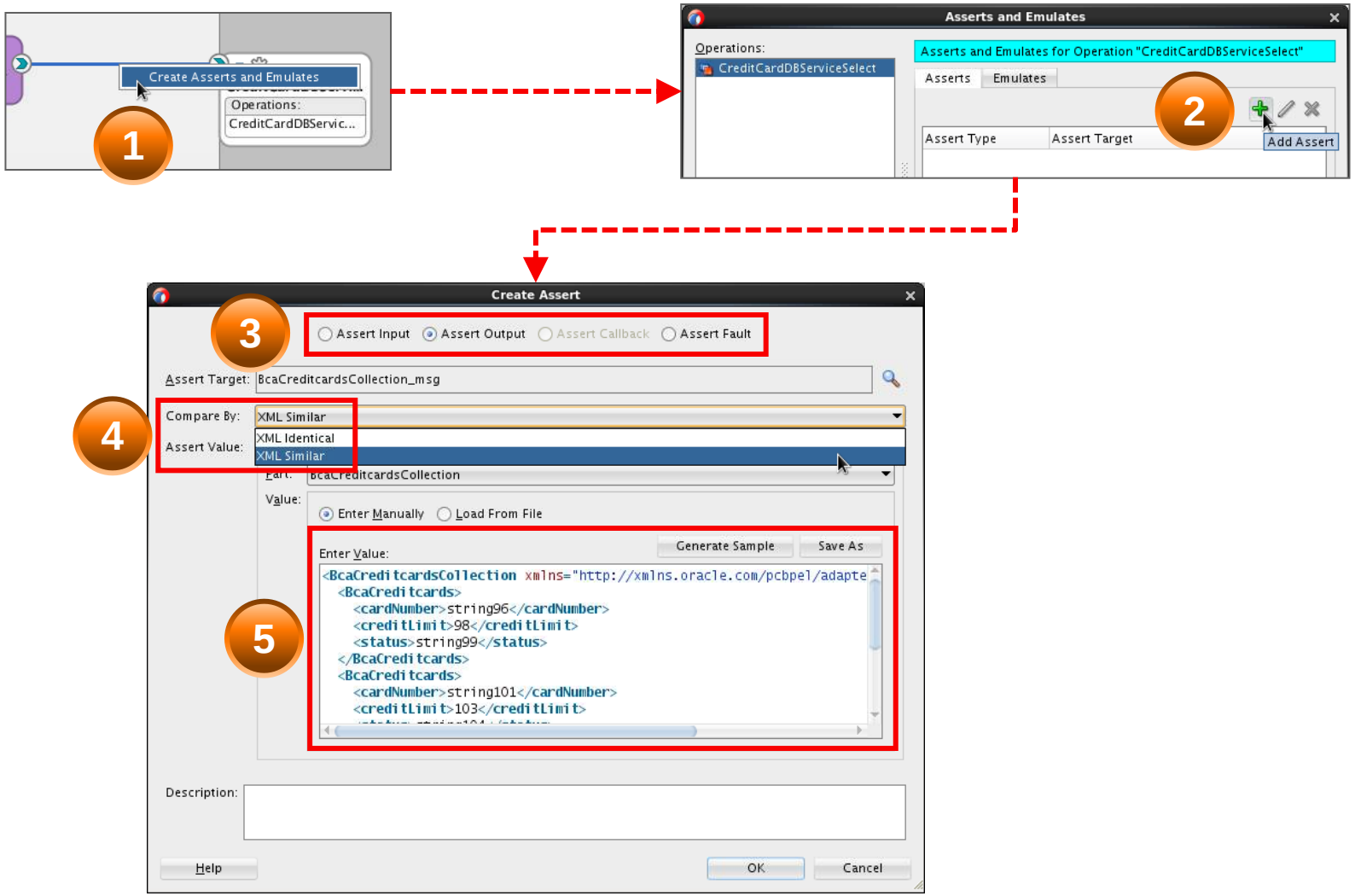
Event messages can be emulated within a test case.



Specify the event to emulate and the target subscribers.

Sample data can be generated and edited.

Creating Assertions



Quiz

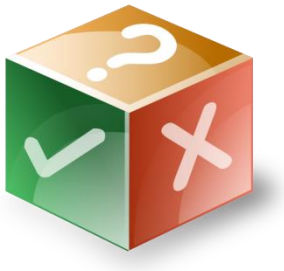
Which of the following must be present for a test case definition to be run as a unit test?

a. Inbound message initiation or an initiating event

b. Outbound message emulation

c. Callback message emulation

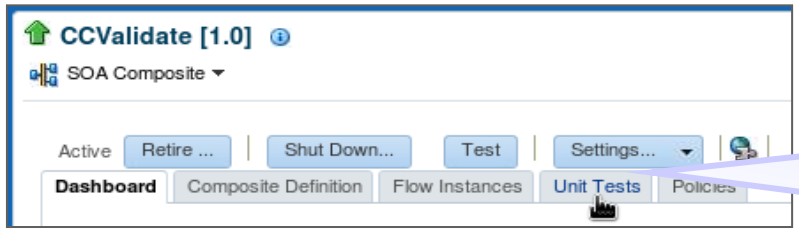
d. Fault message emulation



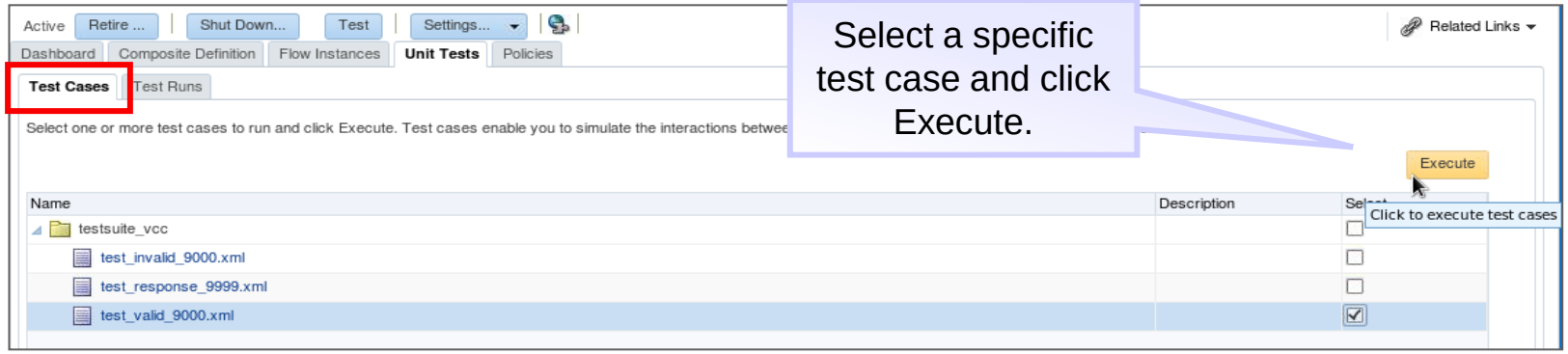
Agenda

- Configuring Test Cases
- Running Test Cases
- Using the SOA Debugger

Selecting the Test Cases to Run

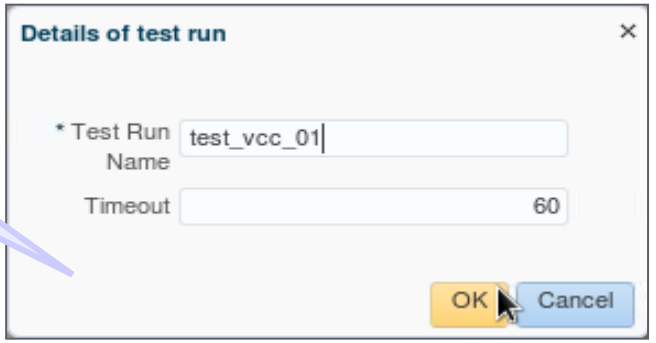


After deploying the application with the test suites, open Enterprise Manager. On the composite application home page, click the Unit Tests tab.



Select a specific test case and click Execute.

Enter a name for the test run and click OK.



Running the Test Cases

Click Test Runs to monitor and view results.

A composite application instance is created for each test case.

Examining Results of a Test Run

Results of Test Run : test_vcc_all (Test Run ID : 160bf8925e9684e5:-50eaeea8:1470fceea42:-7c0d)

Total 3 | Running 0 | Passed 2 | Failed 1 | Unknown 0 | Success Rate 66% | Refresh Test Status

Expand a test suite to view the status of each test case. Click a test suite or test case to view assertion details.

Test suites and test cases	Status
test_suite_vcc	
test_invalid_9000.xml	Passed
test_response_9999.xml	Failed
test_valid_9000.xml	Passed

In the “Assertion details for” section, you can see the list of instance IDs for failed test case composite applications.

Assertion details for test_suite_vcc

☐ Show failures only

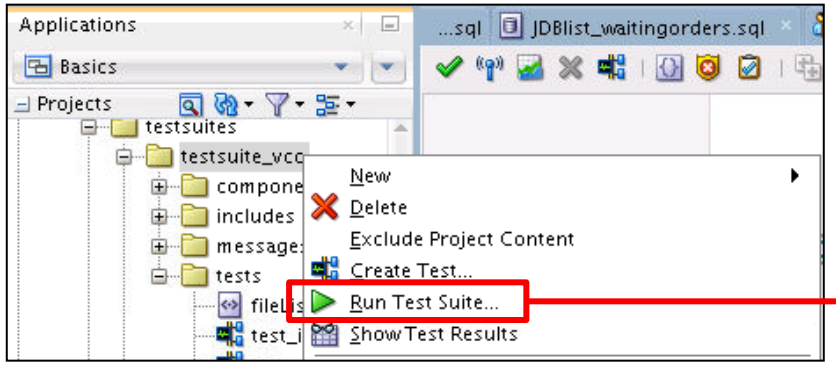
Flow Id	Location	Type	Status	Expected Value	Actual Value	Description	Error Message
40005	ValidateCreditCard_ep	Wire	False	[XML]	[XML]		Expected presence of child nodes to be 'true' but was 'false' - comparing <status...>
40005	ValidateCreditCard_ep	Wire	False	INVALID			Expected <INVALID> but was <>

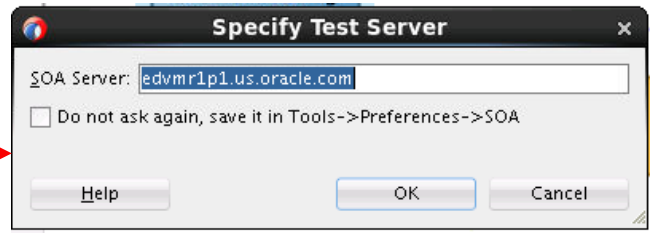
```
<message xmlns="http://xmlns.oracle.com/sca/2006/test"
xmlns:test="http://xmlns.oracle.com/sca/2006/test"> <part
partName="part1"> <content> <CreditCheckResponse
xmlns="http://www.example.org/ns/ccauthorize">
<status>VALID</status> </CreditCheckResponse> </content> </part>
</message>
```

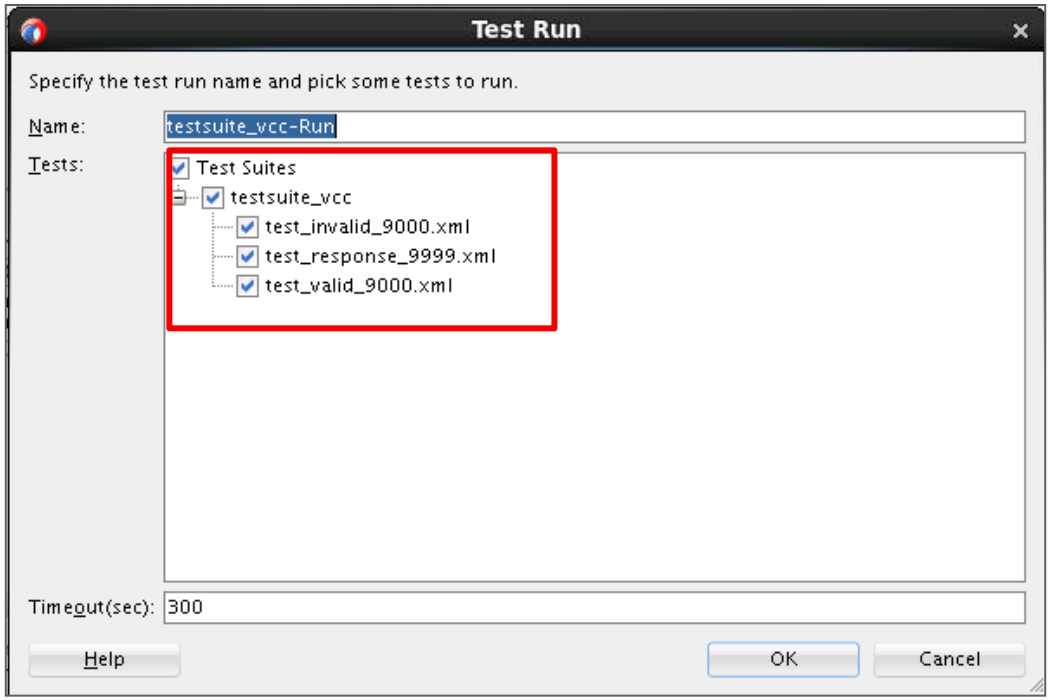
Composite application instance ID link to access the Flow Trace page

Examine the cause of assertion failure.

Running Tests from Within JDeveloper







View Test Run Results

Test Runs

Select a test run to view its test cases below.

Name	Status	Success	Test Cases	Passed	Failed	Errored	Running	Start Time	End Time
test_suite1-Run	✗	50%	2	1	1	0	0	Apr 29, 2012 9:02:10 PM	Apr 29, 2012 9:02:11 PM
test_suite1-Run2	✗	50%	2	1	1	0	0	Apr 29, 2012 9:13:08 PM	Apr 29, 2012 9:13:08 PM
test_suite2-Run	✓	100%	2	2	0	0	0	Apr 29, 2012 9:04:01 PM	Apr 29, 2012 9:04:04 PM
test1-Run	✓	100%	1	1	0	0	0	Apr 29, 2012 8:18:18 PM	Apr 29, 2012 8:18:21 PM
test2-Run	✓	100%	1	1	0	0	0	Apr 29, 2012 8:37:17 PM	Apr 29, 2012 8:37:17 PM

Test Cases: test_suite1-Run2

Select a test case to view its assert results below.

Test	Status	Suite
test1.xml	✓	test_suite1
test2.xml	✗	test_suite1

Assert Results: test2.xml

Click a location to view or edit the assertion in its test editor.

☐ Show Failures Only

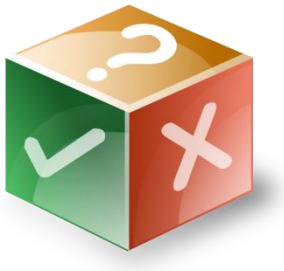
Location	Status	Expected Value	Actual Value	Error Message	Type	Description
AutoLoanService	✓	800	800		Wire	
AutoLoanService	✗	[XML]	[XML]	Expected text value '...	Wire	
AutoLoanService	✓	20,000	20,000		Wire	

Quiz

You start a test suite or test case by clicking the Test button on the composite application home page in Oracle Enterprise Manager.

a. True

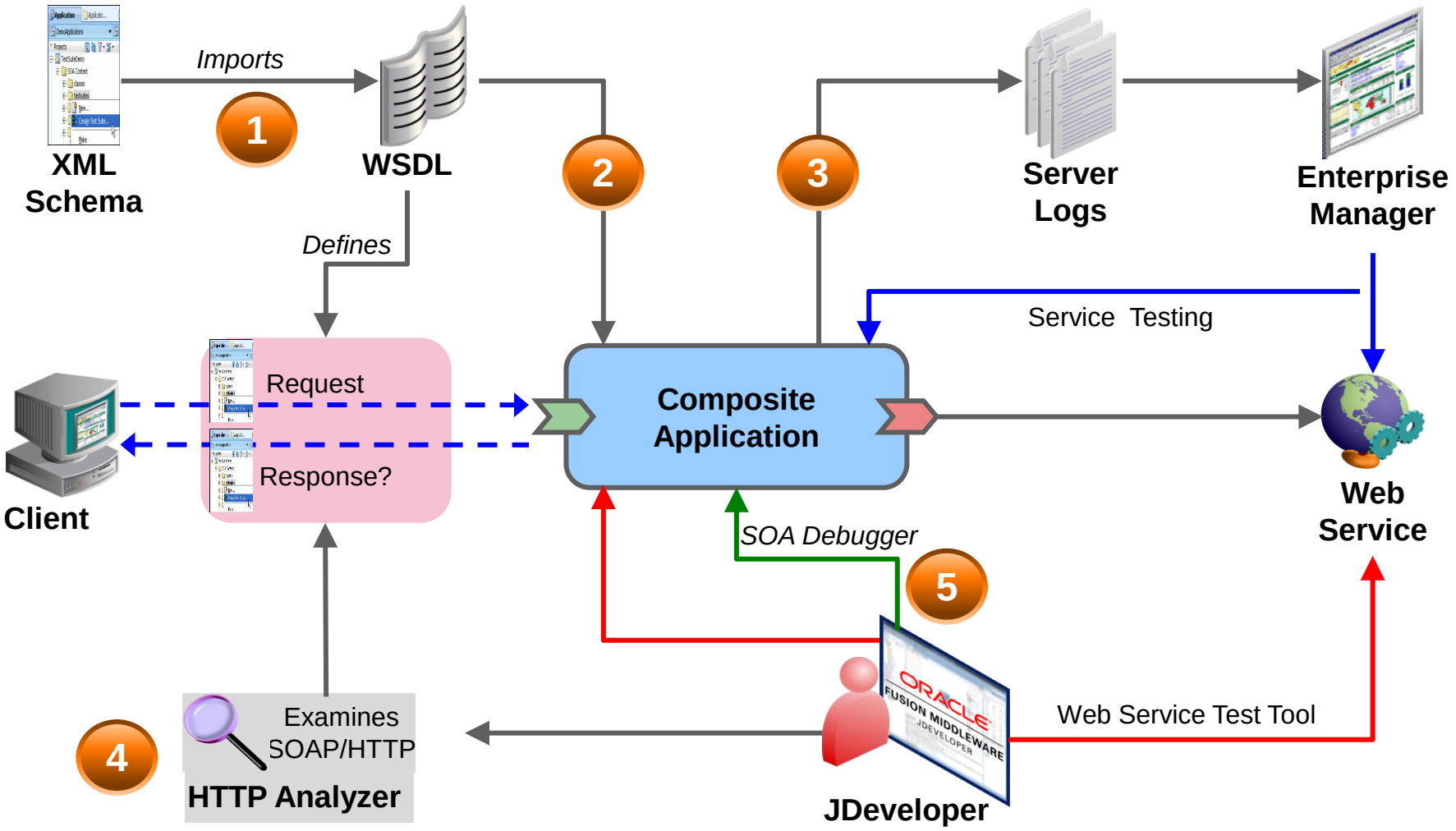
b. False



Agenda

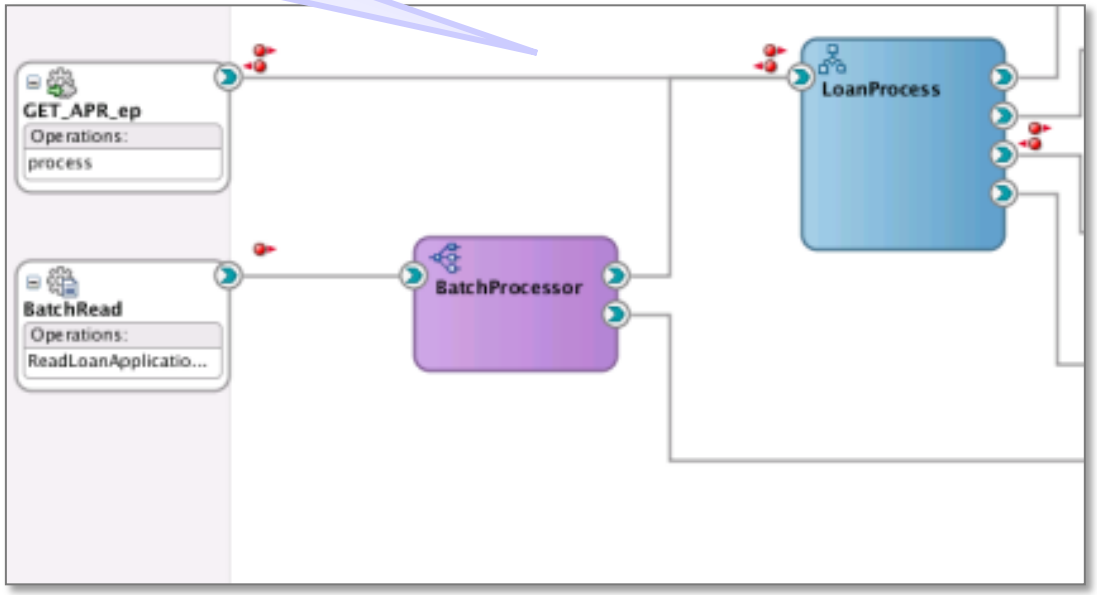
- Configuring Test Cases
- Running Test Cases
- Using the SOA Debugger

Troubleshooting Guidelines



Debugging SOA Composite Applications with the SOA Debugger

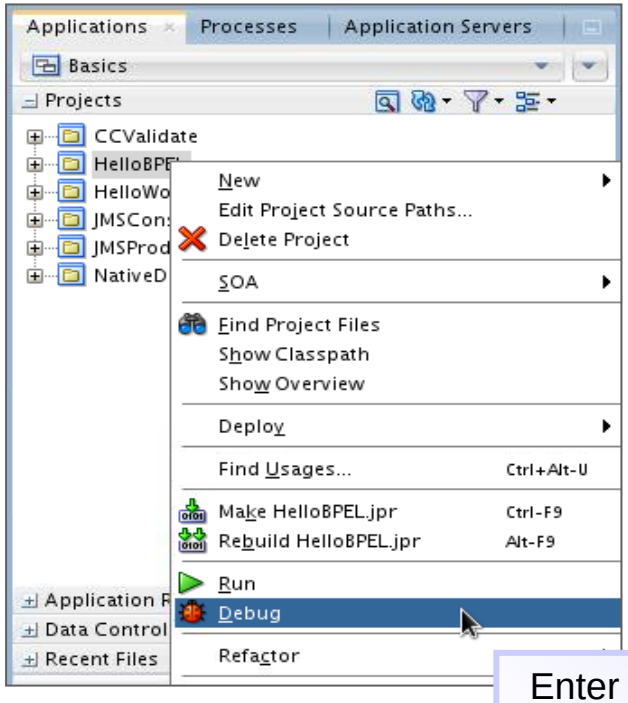
The SOA debugger provides a troubleshooting environment within Oracle JDeveloper.



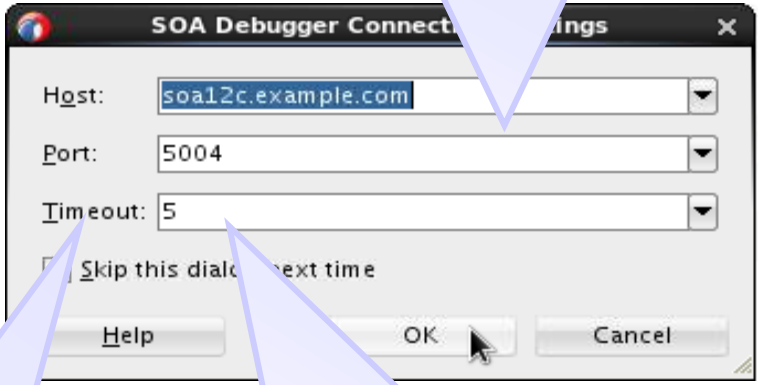
Set breakpoints on binding components, service components, and synchronous and asynchronous BPEL processes.

12c

Starting the Debugger

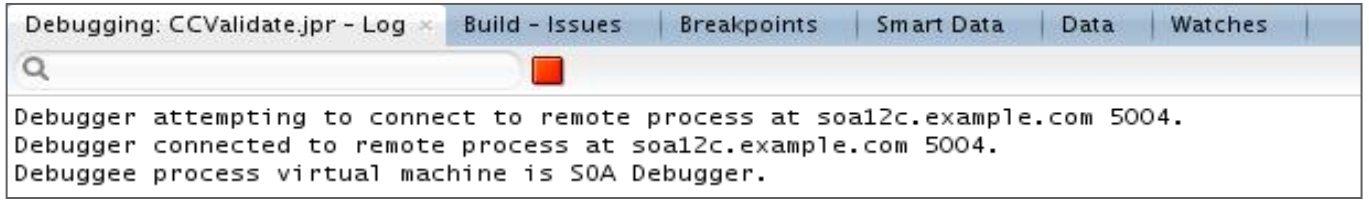


By default, the name of the local host is displayed. You can also enter a remote server.



Enter the port on which the debugging agent listens.

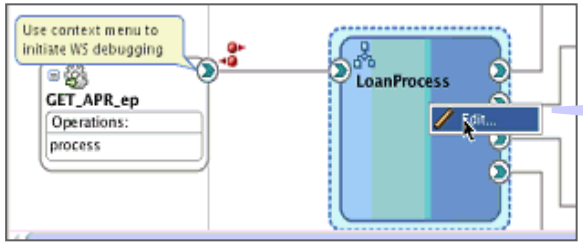
Specify in minutes how long the client should wait while attempting to establish a debugging session before stopping.



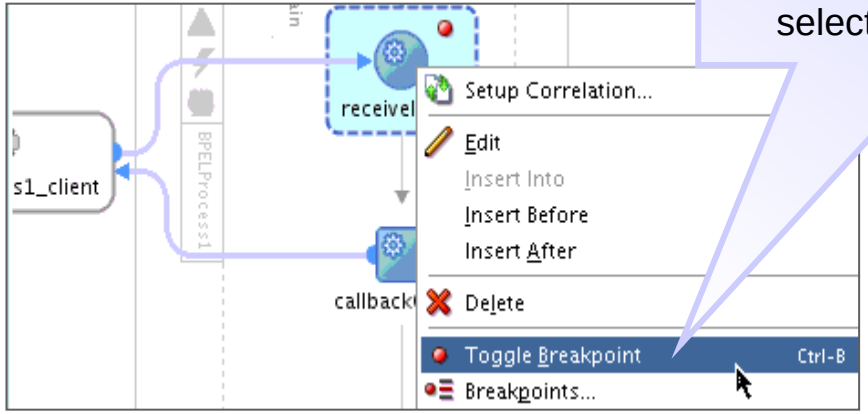
Setting Breakpoints in Services

Choose whether the request, the reply, or both the request and reply are important.

Setting Breakpoints in BPEL

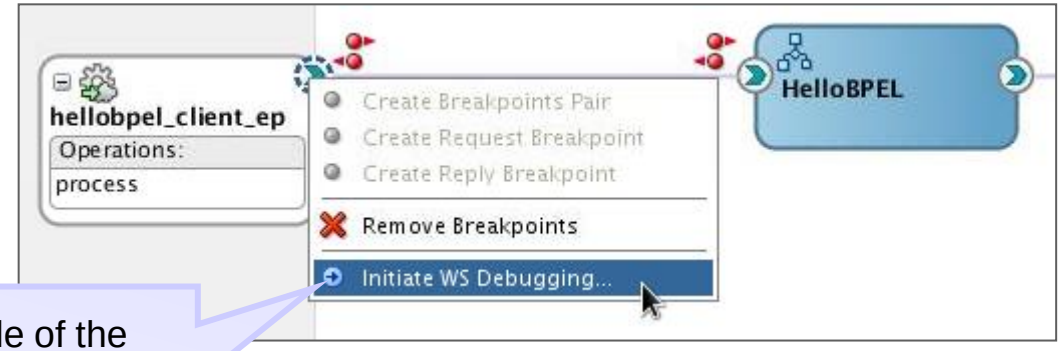


To set a breakpoint on a service component, select Edit.

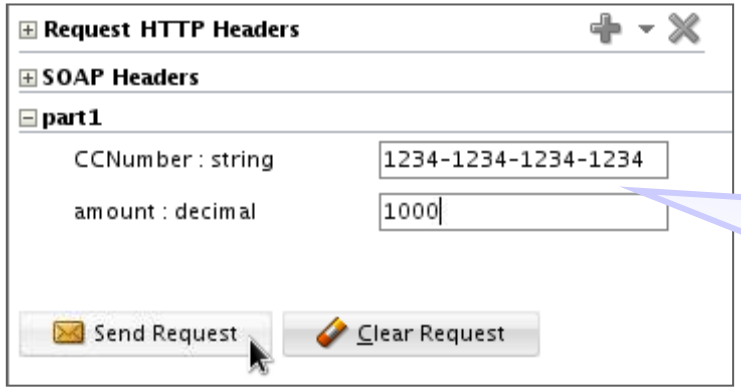


Right-click the BPEL activity on which to set a breakpoint, and select Toggle Breakpoint.

Initiating Debugging



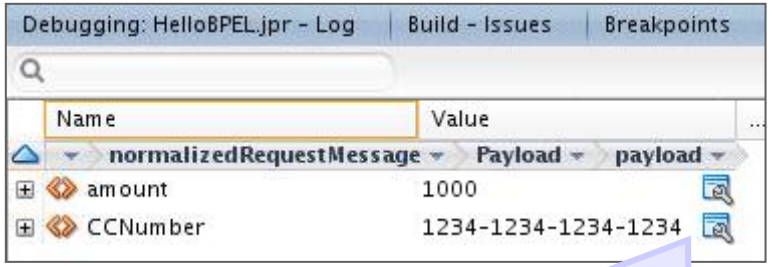
Right-click the right handle of the service binding component and select **Initiate WS Debugging**. This invokes the HTTP Analyzer.



Enter the request message data to send or click HTTP Content to copy and paste the contents from an XML file.
Click Send Request.

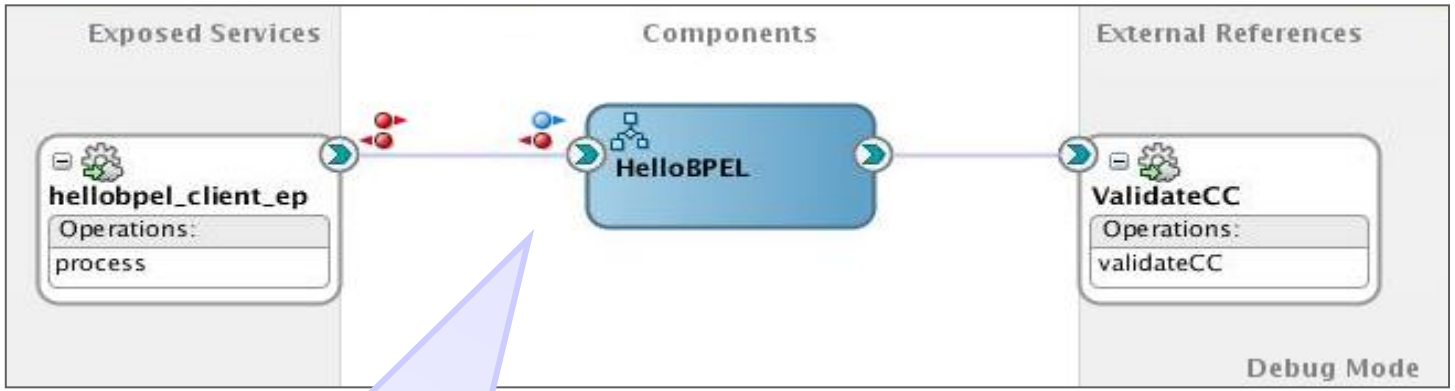
Examining Values

The first breakpoint turns blue and begins pulsing.



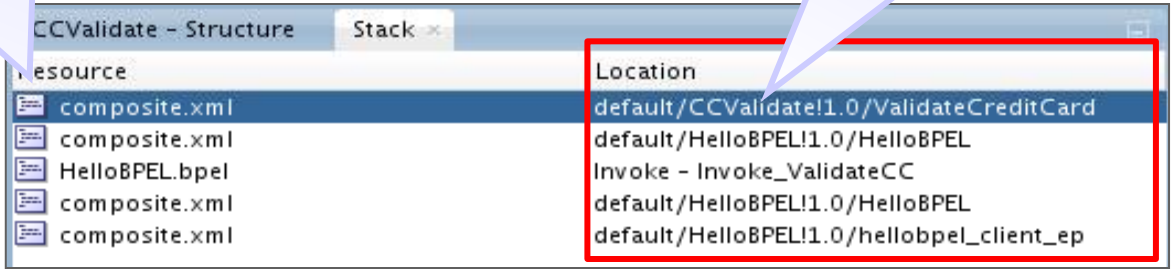
In the Log window at the bottom of Oracle JDeveloper, click Data. Expand the message contents. You can double-click a value to change it.

Frames



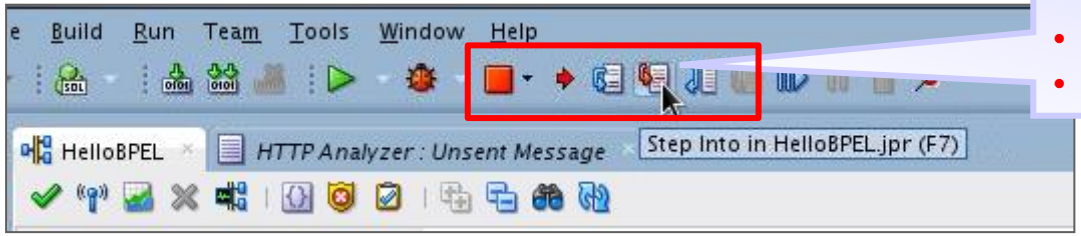
When you create a breakpoint, a corresponding frame is created in the Structure window. A frame is a location.

A stack of frames is a breadcrumb trail of the locations that lead you to your current location.

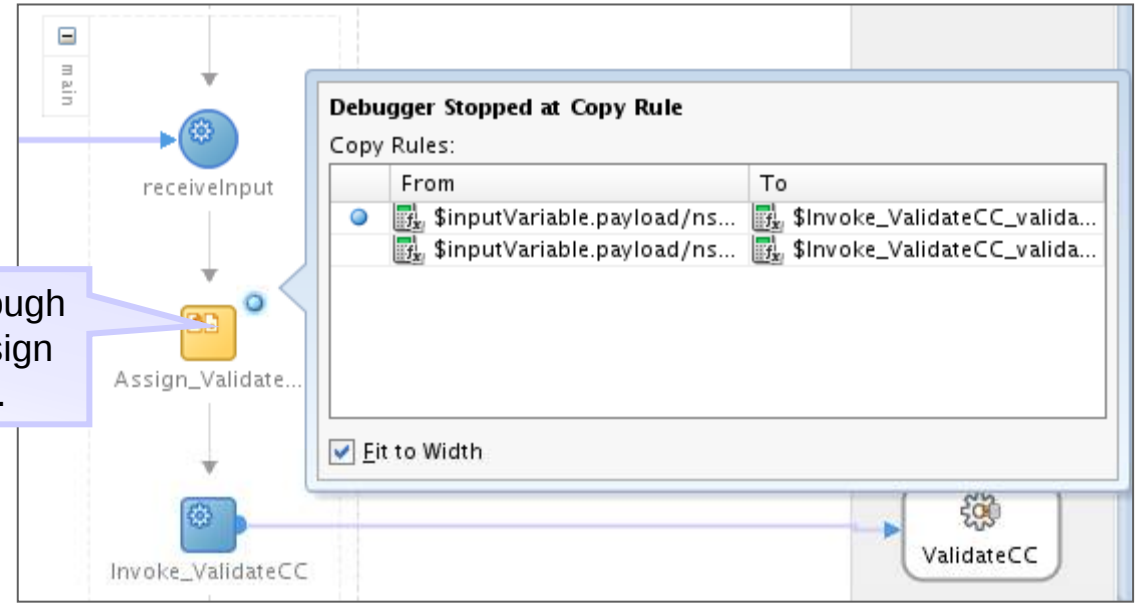


Stepping Through the Application

- F7 Step Into
- Shift-F7 Step Out Of
- Shift-F8 Step Over
- F9 Resume Step



The debugger steps through the copy rules in an Assign activity one at a time.



Ending a Debug Session

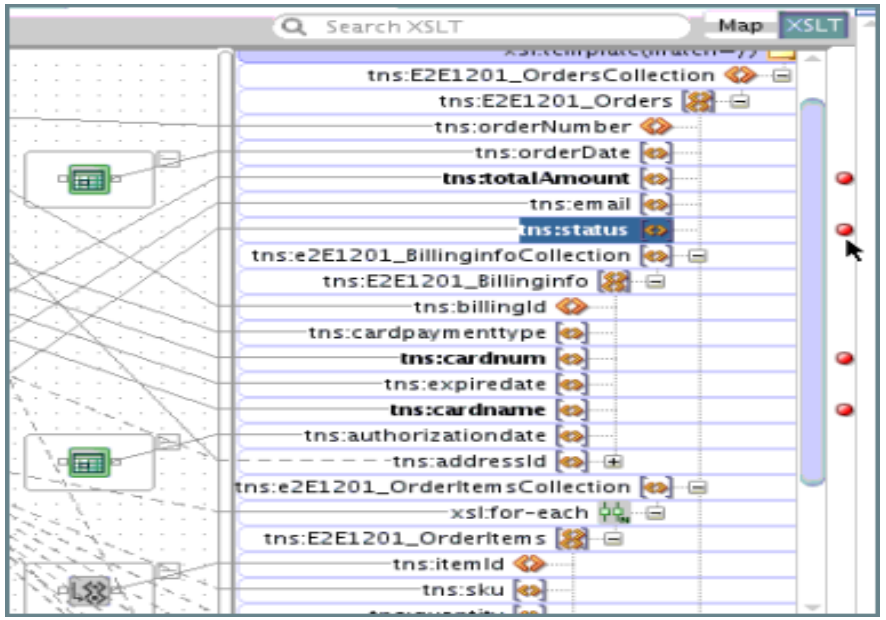
Click the Terminate icon to end a debug session. As long as the debugger is running, the project cannot be edited.

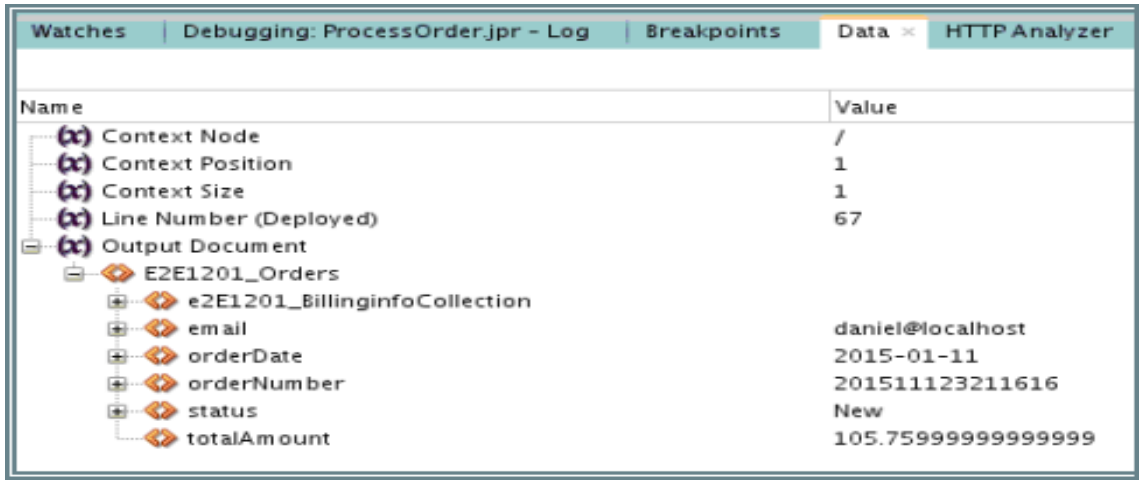
Debugging the XSLT Map

Starting in 12.2.1, you can debug your XSLT maps using the SOA Debugger.

You can debug any XSLT transformation used in a BPEL process or Mediator.

You can also use the debugger with your Oracle Service Bus projects.



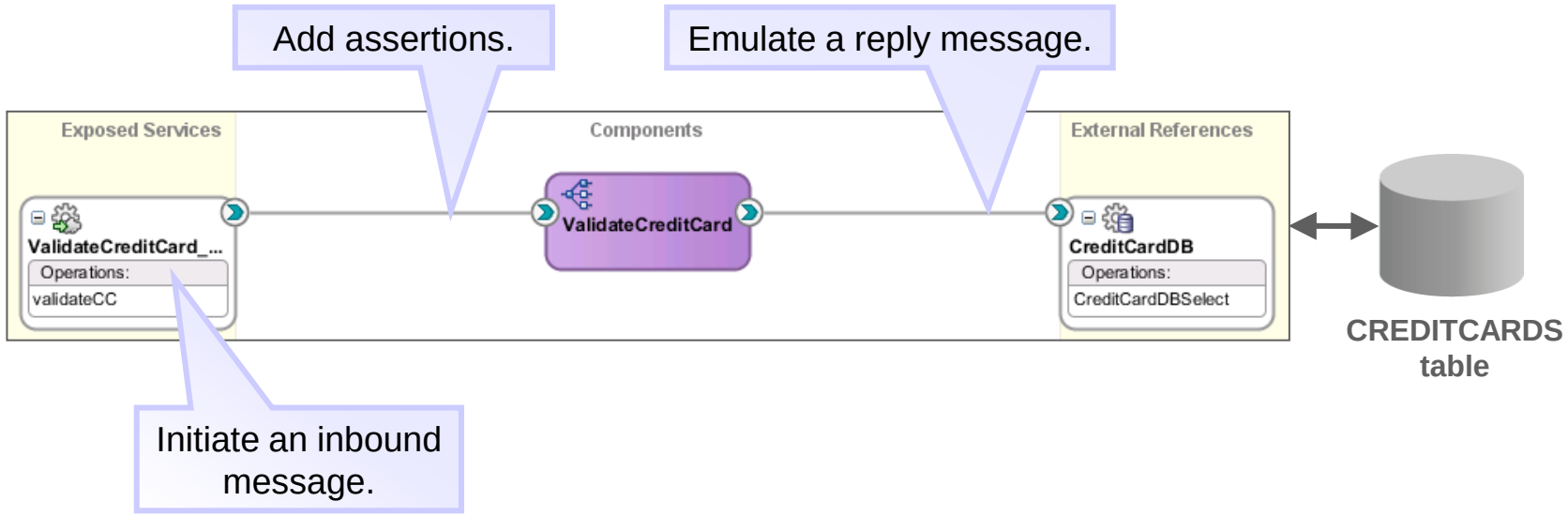


Summary

In this lesson, you should have learned how to:

- Create test suites for composite applications
- Create test cases to initiate inbound messages, and to emulate outbound, fault, and callback messages
- Create test cases with value-based and XML-based assertions
- Discuss strategies for debugging and troubleshooting applications
- Use the SOA debugger to step through an application and observe values during execution

Practice 14-1 to 14-3 Overview



Practice 14-4 Overview

In this practice, you use the SOA debugger to step through a running application.

You observe the values in the HTTP Analyzer.

You step through a running BPEL process, one line at a time, before terminating the debug session.