**Oracle Fusion Order Demo**

The WebLogic Fusion Order Demo application is part of a larger sample application called Fusion Order Demo. In this larger sample application, Global Company sells electronic devices through many channels, including a web-based client application. Electronic devices are sold through a storefront-type web application. Customers can visit the web site, register, and place orders for the products.

**Store Front Module**

The Store Front module provides a rich user interface built with Oracle Application Development Framework to show how to combine an easily built AJAX user interface with a sophisticated SOA composite application. It is based on Oracle ADF business components, ADF model data bindings, and ADF faces.

The Store Front module sells electronic devices through a storefront-type web application.

The Store Front module contains the following projects:

* StoreFrontService: This project provides access to the storefront data and provides transaction support to update data for customers, orders, and products.
* StoreFrontUI: his project provides web pages that the customer uses to browse the storefront, place orders, register on the site, view order information, and update the user profile.

**Opening the Application**

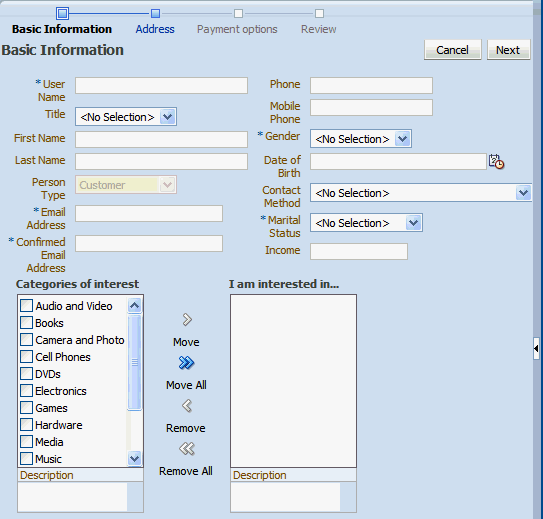
The **StoreFrontService** project contains the classes that allow the product data to be displayed in the web application.

Access the Store Front from the following URL: <http://hostname:port/StoreFrontModule/faces/home.jspx>



**Registration**

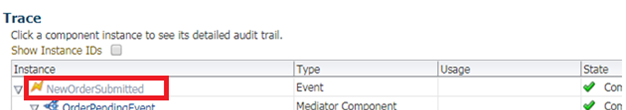
1. When a new customer register then user interface, the web client sends the customer's information to the internal customer service application called **StoreFrontService**.
2. **StoreFrontService** then stores the customer information in a database.
3. The customer can then browse products, add them to their online shopping cart, and place the order.



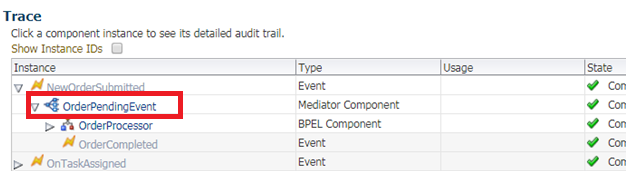
**Placing an order**

After an order is placed, the following sequence occurs to complete the order:

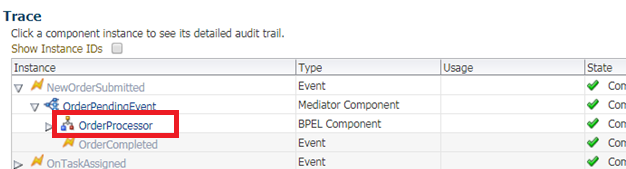
1. Oracle ADF Business Component writes the order to a database with schema for Fusion Order Demo, and raises a **NewOrderSubmitted** event using the Event Delivery Network (EDN). The data associated with this event is the order ID.



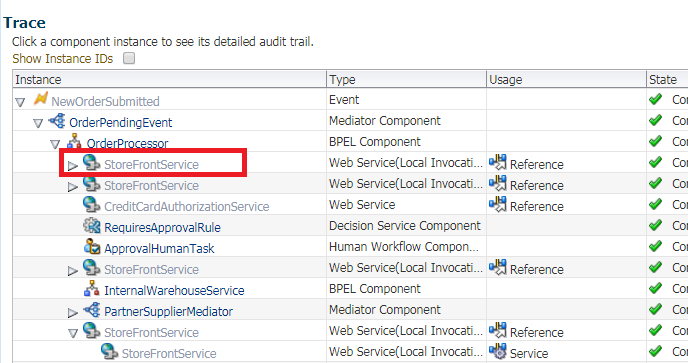
1. Because the **OrderPendingEvent** Oracle Mediator subscribes to the **NewOrderSubmitted** event, the EDN layer notifies the **OrderPendingEvent** Oracle Mediator of the new order.



1. The **OrderPendingEvent** Oracle Mediator receives the order and routes the input order ID to the **OrderProcessor** BPEL process.



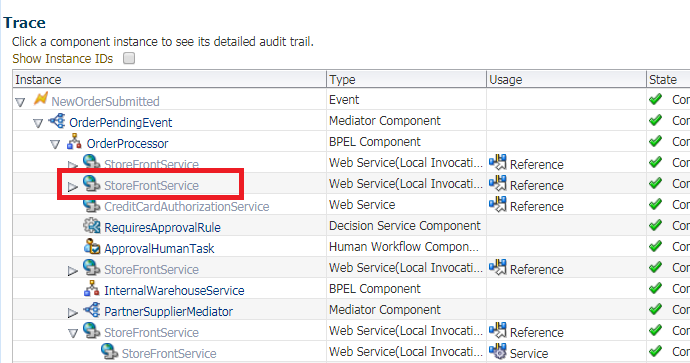
1. The **OrderProcessor** BPEL process receives the order ID from the database, using a bind entity activity to bind to the exposed Oracle ADF Business Component **StoreFrontService** service.



Some of the information about the order used later in the process is:

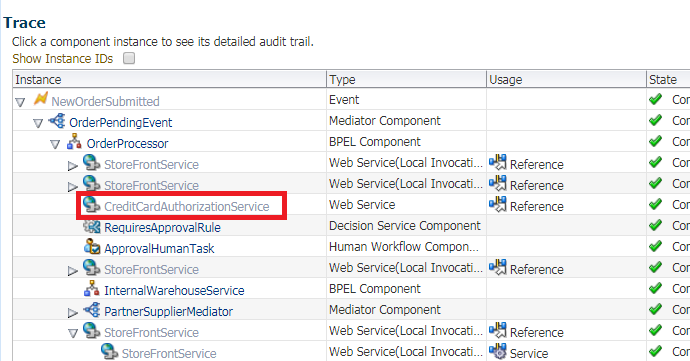
* + Customer ID
  + Items the customer purchased
  + Credit card used
  + Shipping address chosen

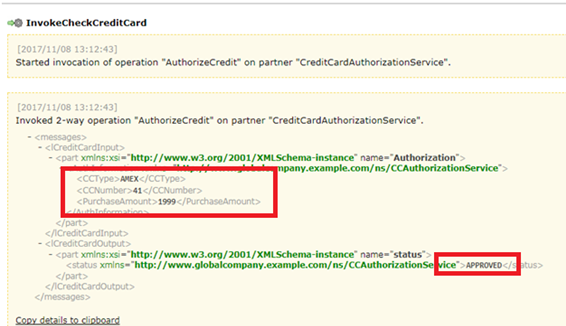
1. The BPEL process initiates **StoreFrontService**, passing it the order ID, to retrieve information about the customer.





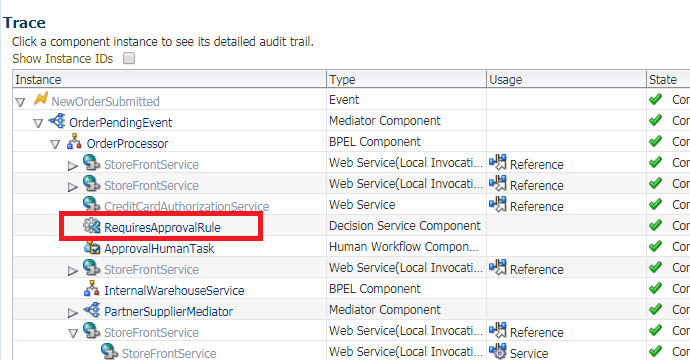
1. The BPEL process then sends the purchase amount, credit card type, and credit card number to **CreditCardAuthorizationService**, which verifies if the customer's credit card is valid.





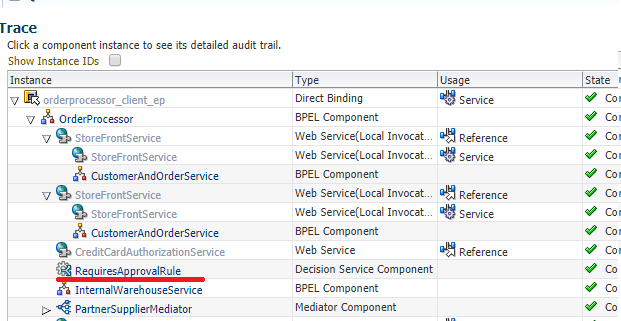
If the credit card is not valid, the BPEL process cancels the order.

If the credit card is valid, the BPEL process sends the order to the **RequiresApprovalRule** business rule to determine if the order requires approval by management.

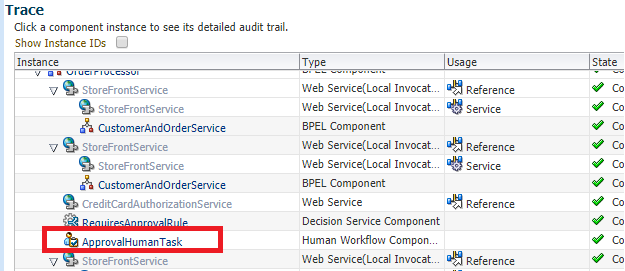


1. The **RequiresApprovalRule** business rule evaluates if manual approval is required.

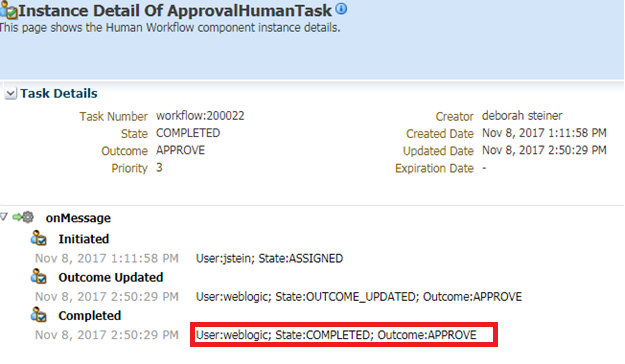
The business rule contains a rule that **does not require manual approval** for orders less than $2,000.



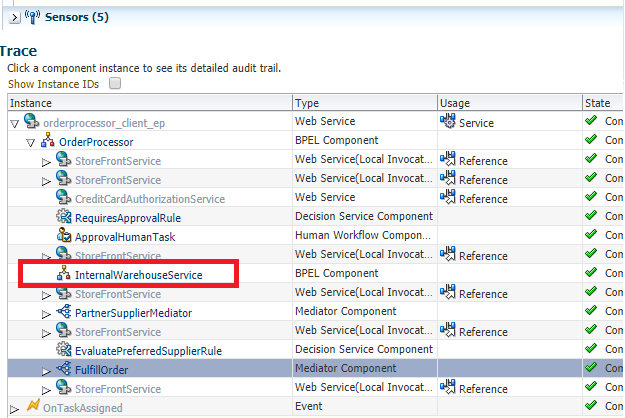
The business rule contains a rule that requires **manual approval** for orders over $2,000.



1. For those orders requiring manual approval, the BPEL process invokes the **ApprovalHumanTask** human task, which in turn performs the following:
   * Routes a message to an assignee named jstein, who then approves or disapproves the order.
   * Publishes the **OnTaskAssigned** event. The **OrderApprovalTaskAssignedMediator** Oracle Mediator subscribes to this event, and if an Oracle BAM Server is configured, it uses an Oracle BAM adapter to send the assignee ID jstein (based on the ECID) of the order to the Oracle BAM Server.

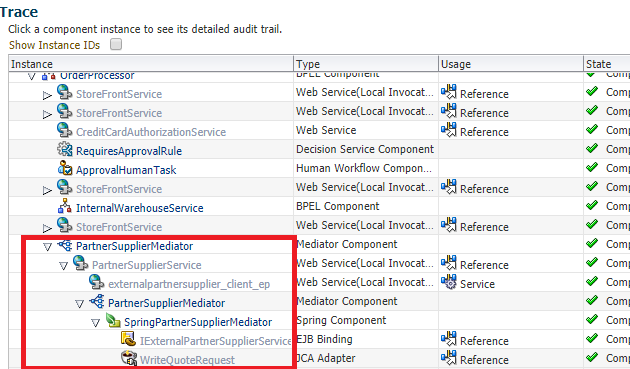


1. If the order is approved, the BPEL process sends the order information to the following suppliers in parallel to obtain a bid:
   * Internal supplier by using the **InternalWarehouseService** BPEL process, also located in **OrderBookingComposite**

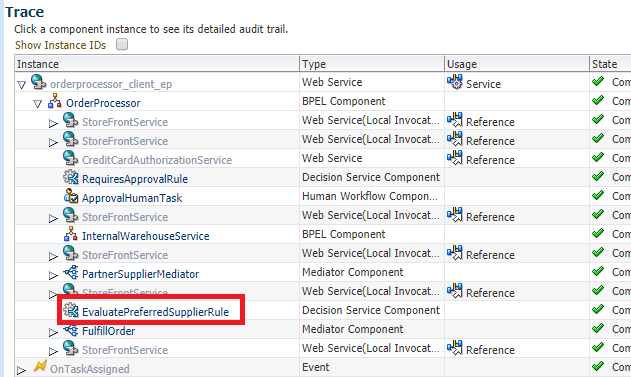




* + External supplier by using the **PartnerSupplierMediator** Oracle Mediator, which in turn routes to the **ExternalPartnerSupplier** BPEL process or **SpringPartnerSupplierMediator** spring component, located in another composite called **PartnerSupplierComposite**

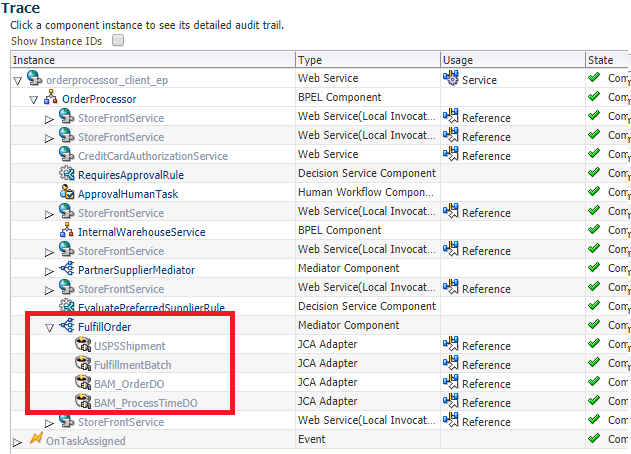


1. The two suppliers respond with their bids, and the BPEL process send the bids to the **EvaluatePreferredSupplierRule** business rule.
2. The **EvaluatePreferredSupplierRule** business rule chooses the supplier with the lower of the two bids.



1. The BPEL process invokes the **FulfillOrder** Oracle Mediator, which performs the following four operations:

* Stores the order in a temporary queue and uploads it to the fulfillment system in batch mode overnight
* Routes the order to USPS
* If an Oracle BAM Server is configured, it uses an Oracle BAM adapter to send data about the order (based on order ID) to the Oracle BAM Server.
* If an Oracle BAM Server is configured, it uses an Oracle BAM adapter to send data about the time for the order to process (based on the instance ID) to the Oracle BAM Server.



1. Once the order is fulfilled, the BPEL process sets the order to **complete**.
2. The BPEL process invokes the **NotificationService** service, which sends the customer an email notification with the purchase order information.
3. When the order completes, the **OrderPendingEvent** Oracle Mediator publishes the **OrderCompleted** business for the **OrderProcessor** process.

