1. ATG-VirtualPiggy Integration
2. User Guide
3. Prepared by:
4. Tarun Gera ([tarun@ace3group.com](mailto:tarun@ace3group.com))
5. Ajmal Mahmood ([ajmal@ace3group.com](mailto:ajmal@ace3group.com))
6. May 8th, 2013
7. Home
8. .
9. Table of Contents

Overview 3

Overview of ATG-Virtual Piggy integration module 3

ATG-Virtual Piggy integration module structure 4

Components and Classes 4

Order related components 5

Order related changes 5

Order Pipeline processing changes in ATG commerce 7

Virtual Piggy Payment related components 10

Payment processing pipeline changes 12

Order Fulfillment related components and changes 12

Profile related components and changes 13

Virtual Piggy Order submission and approval call back components 13

JSP changes in ATG Commerce Reference Store 14

ATG-Virtual Piggy integration Module Configuration 15

/com/virtualpiggy/integration/services/VirtualPiggyServiceConfiguration 15

Purchase Process Configuration 15

/com/virtualpiggy/integration/purchase/VirtualPiggyPurchaseProcessHelper 16

/com/virtualpiggy/integration/purchase/VirtualPiggyShippingInfoDroplet 16

/com/virtualpiggy/integration/profile/VPProfileTools 16

Callback Configuration 16

/com/virtualpiggy/integration/callback/VirtualPiggyCallbackHandler 16

# Overview

The initial deployment and configuration of the ATG-Virtual Piggy integration module is performed by ATG Configuration Installation Manager (CIM). This document describes configuration changes that can be made outside of the CIM process. In addition, this document describes a list of ATG components and classes implemented within the ATG-Virtual Piggy integration module.

# Overview of ATG-Virtual Piggy integration module

ATG-Virtual Piggy integration module enables Virtual Piggy merchants to integrate Virtual Piggy enabled safe E-Commerce with Oracle ATG Web Commerce platform (version: 10.1.2).

This module supports a series of Virtual Piggy application capabilities such as:

1. Checkout flow
   1. Components to handle anonymous child check out flow.
   2. Components to handle Virtual Piggy Token.
2. Child Login flow
   1. Extensions to ATG Login components for Virtual Piggy base authentication.
   2. Extensions to handle Virtual Piggy provided parent information to create/populate ATG shipping groups during checkout flow.
3. ATG Payment System
   1. Extensions for ATG payment system (also called payment groups) to authorize or settle Order payments using Virtual Piggy API.
   2. Integrating Virtual Piggy based payment method with ATG order system and Order submission pipeline processing to populate Virtual Piggy payment during order processing.
4. Order submit and approval Flow
   1. ATG components (also called order pipeline processors) to submit order, to mark order pending approval and to approve order based on Virtual Piggy’s parent approval flow.
   2. Update the order submission status if successful.
5. ATG Order Update Callback URL
   1. Update the order status appropriately based on the response of the payment transaction.
   2. If success, move the order to fulfillment process.

There are two types of Merchant supported with this integration module:

* One Step merchants: Order will be authorized and settled within a single transaction during order submission. In the case of child checkout flow when a parent’s approval is required, the settlement of the order will occur after receiving the approval from a parent.
* Two Step merchants: Oder will be authorized first during order submission. The settlement of the order will be after the order is shipped/completed. Two steps merchants have the option to capture the payment themselves, or they could choose Virtual Piggy to capture the payment too.

# ATG-Virtual Piggy integration module structure

This section elaborates files included in the integration module,

Module structures are shown as follows:

|  |
| --- |
| VirtualPiggyCommerce/  Src/  cim - All CIM wizard files  config - All component configuration files  crsJspFiles - These are jsp files that override ATG Commerce Reference Store jsp files  java - Java source code  lib – Third party libraries  META-INF - Module manifest file  sql - Sql files for extension into ATG schemas  wsdl - Virtual Piggy transaction service WSDL file |

Build scripts included inthe integration module support the following ant targets

clean: Cleans up build directory

install: compiles src and install module at defined %DYNAMO\_HOME% location

generate-stubs: generates java web service client stub classes for SOAP transaction service

All: clean up build and deploy directories, generate web services stubs, compiles source code and install the code to ATG installation location.

# Components and Classes

This section elaborates the key components within ATG-Virtual Piggy integration module.

## Order related components

Order related components are created or extended to support the checkout flow for a Virtual Piggy order.

**ProcCheckVirtualPiggyOrderState**

|  |  |
| --- | --- |
| Component Name | /com/virtualpiggy/integration/order/processor/CheckVirtualPiggyOrderState |
| Class | com.virtualpiggy.integration.order.processor.ProcCheckVirtualPiggyOrderState |
| Key function | This processor is added to ATG Process Order pipeline to determine if the order's state is pending for parent approval and if it requires a call back from Virtual Piggy. If it is true then it delays the process of sending order to fulfillment until call back from Virtual Piggy is successful. |
| Component configuration | orderStates=/atg/commerce/states/OrderStates |

## Order related changes

The following section presents changes made in ATG related repository definition and components

1. **Component: /atg/commerce/order/OrderTools**

The Order Tools component controls many aspects of the purchase process such as mapping between commerce object types and class names, defining the default commerce object types, and mapping between commerce objects and item descriptors

|  |
| --- |
| paymentTypeClassMap+=\  virtualPiggyPayment=com.virtualpiggy.integration.order.payment.VirtualPiggyPayment  beanNameToItemDescriptorMap+=\  com.virtualpiggy.integration.order.payment.VirtualPiggyPayment=virtualPiggyPayment,\  com.virtualpiggy.integration.order.payment.VirtualPiggyPaymentStatus=virtualPiggyPaymentStatus |

***paymentTypeClassMap -*** maps Virtual Piggy payment group to class com.virtualpiggy.integration.order.payment.VirtualPiggyPayment

***beanNameToItemDescriptorMap*** - maps customer order repository extensions of item types

***VirtualPiggyPayment*** and ***VirtualPiggyPaymentStatus*** are two java beans which act as getter/setters for persistence or order in ATG order repository.

1. **Repository Definition /atg/commerce/order/orderrepository.xml**

Order repository GSA template file defines how ATG orders are persistent in database and does SQL generation and conversion/mapping of java objects (ATG types) into Relational objects (database tables).

The order repository definition extension:

|  |
| --- |
| <item-descriptor name="virtualPiggyPayment" sub-type-value="virtualPiggyPayment" display-name="virtualPiggyPayment"  use-id-for-path="false" super-type="paymentGroup" hidden="false" expert="false" content="false" writable="true"  cache-mode="simple" default="false" item-cache-timeout="180000" id-separator=":">  <attribute name="resourceBundle" value="atg.commerce.OrderRepositoryTemplateResources" />  <table name="virtualpiggy\_payment" type="auxiliary" id-column-name="payment\_group\_id">  <property name="vpToken" column-name="VP\_TOKEN" data-type="string" display-name="Virtual Piggy Token"/>  <property name="vpChildId" column-name="VP\_CHILD\_ID" data-type="string" display-name="Virtual Piggy ChildId"/>  <property name="vpTransDesc" column-name="VP\_TRANS\_DESC" data-type="string" display-name="Virtual Piggy Transaction Description" />  <property name="vpPaymentAccId" column-name="VP\_PAYMENT\_ACCOUNT\_ID" data-type="string" display-name="Virtual Piggy Payment Account Id" />  </table>    </item-descriptor>  <item-descriptor name="virtualPiggyPaymentStatus" super-type="paymentStatus" sub-type-value="virtualPiggyPaymentStatus">  <table name="virtualpiggy\_payment\_status" type="auxiliary" id-column-name="status\_id">  <property name="vpTransactionStatus" data-type="string" column-name="VP\_TRANSACTION\_STATUS" display-name="Virtual Piggy Transaction Status">  <!-- attribute name="useCodeForValue" value="false" />  <option value="Processed" code="0" />  <option value="ApprovalPending" code="1" />  <option value="LimitsExceeded" code="2" />  <option value="Error" code="3" /-->  </property>  </table>  </item-descriptor> |

**VirtualPiggyPayment** item type is extension of ATG payment type and it adds following properties

* VirtualPiggyToken
* VirtualPiggyChildID
* VirtaulPiggyTransactionDescription
* vpPaymentAccId

These properties are saved in db table virtualpiggy\_payment

VirtualPiggyPaymentStatus item type stores Payment Status as

* Processed
* PendingApproval
* LimitExceded
* Error status

## Order Pipeline processing changes in ATG commerce

Virtual Piggy integration module appends the following pipeline link to the existing ATG Commerce Pipeline

**Pipeline Chain Process Order**

This is called pipeline chain is invoked when order is submitted on order confirmationpage.

Following is changed pipeline

|  |
| --- |
| <pipelinechain name="processOrder" xml-combine="append">  <pipelinelink name="sendPromotionUsedMessage" transaction="TX\_MANDATORY" xml-combine="replace">  <processor jndi="/atg/commerce/order/processor/SendPromotionUsedMessage"/>  <transition link="virtualPiggyConditionalSendToFulfillment" returnvalue="1"/>  </pipelinelink>  <pipelinelink name="virtualPiggyConditionalSendToFulfillment" transaction="TX\_MANDATORY">  <processor jndi="/com/virtualpiggy/integration/order/processor/CheckVirtualPiggyOrderState"/>  <transition link="sendFulfillmentMessage" returnvalue="1"/>  <transition link="skipFulfillmentMessage" returnvalue="2"/>  </pipelinelink>  <pipelinelink name="skipFulfillmentMessage" transaction="TX\_MANDATORY">  <processor class="atg.service.pipeline.processor.EmptyProcessor"/>  </pipelinelink>  </pipelinechain> |

These changes prevent order to be submitted to ATG fulfillment if order contain virtual piggy related payment group. Order is submitted to ATG to ATG fulfillment only after call back processing. Class **com.virtualpiggy.integration.callback.VirtualPiggyCallbackHandler**

Then invokes following pipeline to chain to send order for fulfillment processing

|  |
| --- |
| <pipelinechain name="vpProcessOrder" transaction="TX\_REQUIRED" headlink="sendFulfillmentMessage2">  <pipelinelink name="sendFulfillmentMessage2" transaction="TX\_MANDATORY">  <processor jndi="/atg/commerce/order/processor/SendFulfillmentMessage"/>  </pipelinelink>  </pipelinechain> |

Pipeline Chain **validatePaymentGroup** is invoked from **paymentgroupformHandler** servlet bean on payment processing jsp page. It is defined as following

|  |
| --- |
| <pipelinechain name="validatePaymentGroup">  <pipelinelink name="dispatchOnPGType">  <transition returnvalue="11" link="validateVirtualPiggyPayment"/>  </pipelinelink>  <pipelinelink name="validateVirtualPiggyPayment">  <processor jndi="/com/virtualpiggy/integration/order/payment/processor/ValidateVirtualPiggyPayment"/>  </pipelinelink>  </pipelinechain>  <pipelinechain name="validatePaymentGroupPreConfirmation">  <pipelinelink name="dispatchOnPGTypePreConfirmation">  <transition returnvalue="11" link="validateVirtualPiggyPaymentPreConfirmation" />  </pipelinelink>  <pipelinelink transaction="TX\_MANDATORY" name="validateVirtualPiggyPaymentPreConfirmation">  <processor jndi="/com/virtualpiggy/integration/order/payment/processor/ValidateVirtualPiggyPayment"/>  </pipelinelink>  </pipelinechain> |

The pipeline link **dispatchOnPGType** returns 11 if payment group if of type virtual piggy payment group type which forces additional verification of payment group via **ValidateVirtualPiggyPayment**  which validates if ChildID and Authorization Token field have valid data before payment group is added to order

Same validation is performed before order is confirmed on order confirmation page via pipeline chain **validatePaymentGroupPreConfirmation**

Pipeline Chain **validatePaymentGroupNoApproval** is defined as following:

|  |
| --- |
| <pipelinechain name="validatePaymentGroupNoApproval">  <pipelinelink name="dispatchOnPGTypeNoApproval">  <transition returnvalue="11" link="validateVirtualPiggyPaymentNoApproval" />  </pipelinelink>  <pipelinelink transaction="TX\_MANDATORY" name="validateVirtualPiggyPaymentNoApproval">  <processor class="atg.service.pipeline.processor.EmptyProcessor"/>  </pipelinelink>  </pipelinechain> |

The above pipeline chain is designed to skip the ATG OOTB Approval Pipeline.

## Virtual Piggy Payment related components

Virtual piggy integration module introduces a new payment type “VirtualPiggyPayment” in ATG commerce framework. This section describes key component

**PaymentManager**

|  |  |
| --- | --- |
| Component Name | /atg/commerce/payment/PaymentManager |
| Class | com.virtualpiggy.integration.order.payment.VirtualPiggyPaymentManager |
|  | This OOTB Component is extended with VirutlaPiggyPaymentManager class.  It overrides postProcesAuthorize method for settling the payment if merchant type is One Step and there are no callback approvals required. |
| Component configuration | paymentGroupToChainNameMap+=com.virtualpiggy.integration.order.payment.VirtualPiggyPayment=virtualPiggyPaymentProcessorChain  VPConfiguration=/com/virtualpiggy/integration/services/VirtualPiggyServiceConfiguration  paymentGroupStates=/atg/commerce/states/PaymentGroupStates |

**VirtualPiggyPaymentProcessor**

|  |  |
| --- | --- |
| Component Name | /com/virtualpiggy/integration/order/payment/VirtualPiggyPaymentProcessor |
| Class | com.virtualpiggy.integration.order.payment.processor.VirtualPiggyPaymentProcessorImpl |
|  | This component is Virtual Piggy payment processor which using Transaction web service to do following   1. Authorize - This method uses Order’s payment info object (of type VirtualPiggyPaymentInfo) and makes payment authorization call and on successful authorization updates VirtualPiggyPaymentStatus bean which is serialized and saved in order repository 2. Debit - This method does settlement of Virtual Piggy order and if merchant is configured to perform capture transaction, ATG will complete order without capture of the payment. 3. Credit - (currently stub method) |
| Component configuration | service=/com/virtualpiggy/integration/services/VirtualPiggyIntegrationService  successTrStatuses=Processed,ProcessingPending,ApprovalPending |

**VirtualPiggyPaymentGroupFormHandler**

|  |  |
| --- | --- |
| Component Name | /com/virtualpiggy/integration/purchase/ VirtualPiggyPaymentGroupFormHandler |
| Class | com.virtualpiggy.integration.purchase. VirtualPiggyPaymentGroupFormHandler |
| Key function | This component is extension of payment group form handler and adds empty payment group of type VirtualPiggyPaymentGroup into payment container before payment processing |
| Component configuration | $scope=request  configuration=/atg/commerce/order/purchase/PurchaseProcessConfiguration  order^=/atg/commerce/ShoppingCart.current  profile=/atg/userprofiling/Profile  validatePaymentGroupsChainId=moveToConfirmation  applyDefaultPaymentGroup=true  allowPaymentGroupsWithZeroAmount=false |

**VirtualPiggyPaymentInfoDroplet**

|  |  |
| --- | --- |
| Component Name | /com/virtualpiggy/integration/purchase/ VirtualPiggyPaymentInfoDroplet |
| Class | com.virtualpiggy.integration.purchaseVirtualPiggyPaymentInfoDroplet |
| Key function | This droplet is used on the payment page to show payment options (for parent user) or use VP account associated default payment (for child user). For parent user, payment list is retrieved from VP parent account and set on the profile. Call is made to VP system if the information is not already retrieved. Otherwise it is retrieved from the profile. Sets output oparam for parent user. Sets default oparam for child user. Sets error oparam for any errors while processing the request. |
| Component configuration | $scope=request  profile=/atg/userprofiling/Profile  virtualPiggyPurchaseProcessHelper=/com/virtualpiggy/integration/purchase/VirtualPiggyPurchaseProcessHelper  vpTools=/com/virtualpiggy/integration/profile/VPProfileTools |

## Payment processing pipeline changes

Payment Processing Pipeline is invoked by PaymentManager for **VirtualPiggyPayment** PaymentGroup. Please find the extended Commerce Pipeline definition file as follows:

|  |
| --- |
| <pipelinemanager>  <!-- This chain is used to process Virtual Piggy payment group to authorize/credit/debit a payment group.-->  <pipelinechain name="virtualPiggyPaymentProcessorChain" transaction="TX\_REQUIRED" headlink="createVirtualPiggyPaymentInfo">  <pipelinelink name="createVirtualPiggyPaymentInfo" transaction="TX\_MANDATORY">  <processor jndi="/com/virtualpiggy/integration/order/payment/processor/CreateVirtualPiggyPaymentInfo"/>  <transition returnvalue="1" link="processVirtualPiggyPayment"/>  </pipelinelink>  <pipelinelink name="processVirtualPiggyPayment" transaction="TX\_MANDATORY">  <processor jndi="/com/virtualpiggy/integration/order/payment/processor/ProcessVirtualPiggyPayment"/>  </pipelinelink>  </pipelinechain>  </pipelinemanager> |

This **virtualPiggyPaymentProcessorChain** does payment settlement via **/com/virtualpiggy/integration/order/payment/processor/ProcessVirtualPiggyPayment** processor.

## Order Fulfillment related components and changes

ATG Virtual Piggy Integration Module allows payment settlement as soon as it is authorized if the merchant type is One Step.

**SettleOrder**

|  |  |
| --- | --- |
| Component Name | /atg/commerce/fulfillment/processor/SettleOrder |
| Class | com.virtualpiggy.integration.order.processor.ProcVirtualPiggySettleOrder |
| Key function | This component will bypass OOTB settlement logic if the merchant type is One Step and Payment is already settled. |
| Component configuration | VPConfiguration=/com/virtualpiggy/integration/services/VirtualPiggyServiceConfiguration |

## Profile related components and changes

ATG-Virtual Piggy integration module allows anonymous customer to get authenticated via following component

**ProfileFormhandler**

|  |  |
| --- | --- |
| Component Name | /atg/userprofiling/ProfileFormhandler |
| Class | com.virtualpiggy.integration.profile.VirtualPiggyProfileFormHandler |
| Key function | This component authenticate user , check if user is authenticated, get Children for parent user ,select and validate child selection |
| Component configuration | virtualPiggyUserNameParam=vpUserName  virtualPiggyPasswordParam=vpPassword  vpTools=/com/virtualpiggy/integration/profile/VPProfileTools |

## Virtual Piggy Order submission and approval call back components

During order processing, this integration module submits an order for approval and depending upon approval process set by parent user, a call back from Virtual Piggy gateway is invoked. ATG Commerce Reference Store demonstrates how to set up a Call back listener servlet .

Following is configuration for Callback servlet in web.xml

|  |
| --- |
| <servlet>  <servlet-name>VirtualPiggyCallbackServlet</servlet-name>  <servlet-class>com.virtualpiggy.integration.callback.VirtualPiggyCallbackServlet</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>VirtualPiggyCallbackServlet</servlet-name>  <url-pattern>/servlets/virtualPiggyCallbackServlet</url-pattern>  </servlet-mapping> |

VirtualPiggyCallbackServlet invokes a nucleus component

**VirtualPiggyCallBackHandler**on each call back invocation. Following is configuration for Callback handler.

|  |  |
| --- | --- |
| Component Name | /com/virtualpiggy/integration/callback/VirtualPiggyCallbackHandler |
| Class | com.virtualpiggy.integration.callback.VirtualPiggyCallbackHandler |
| Key function | On each call back, this component parses call back data to retrieve order ID and validates if Virtual Piggy payment transaction is successful.  If order is approved, the order status is updated as “SUBMITTED” and sent for fulfillment process. It also settles the payment if the merchant type is One Step.  If order is not approved, the order status is updated as “REMOVED”.  If there is any error or exception, order status is updated as “PENDING\_MERCHANT\_ACTION.This order has to be manually corrected and resend. |
| Component configuration | orderStates=/atg/commerce/states/OrderStates  paymentGroupStates=/atg/commerce/states/PaymentGroupStates  VPConfiguration=/com/virtualpiggy/integration/services/VirtualPiggyServiceConfiguration  successStatuses=Processed  rejectStatuses=Rejected  # everything else is error.  orderManager=/atg/commerce/order/OrderManager  pipelineManager=/atg/commerce/PipelineManager  clientLockManager=/atg/dynamo/service/ClientLockManager  transactionLockFactory=/atg/commerce/util/TransactionLockFactory |

## JSP changes in ATG Commerce Reference Store

The following table describes JSP changes required in ATG Commerce Reference Store application to demonstrate the ATG-Virtual Piggy purchase flow. These files are copied to store.war file by CIM.

|  |  |
| --- | --- |
| File Name | description |
| /checkout/login.jsp | Displays additional Virtual piggy login option for parent and child users. |
| /checkout/Billing.jsp | Detects if user is authenticated via virtual piggy then instead of collecting billing info it displays  “Payment Account associated with your Virtual Piggy Profile will be used to pay this order.” |
| /checkout/Shipping.jsp | Detects if user is authenticated via virtual piggy then retrieves and displays shipping address from virtual piggy |
| /checkout/SelectVirtualPiggyChild.jsp | For parent user , allows selection of which for which order is being placed. |
| /global/gadgets/odersingleShippingInfo.jsp | Modified to prevent option to edit ship/bill addresses in case of virtual piggy order |

# ATG-Virtual Piggy integration Module Configuration

This section outlines the related configuration required for ATG-Virtual Piggy integration module:

### /com/virtualpiggy/integration/services/VirtualPiggyServiceConfiguration

**merchantId**–this property specifies the merchant id provided by Virtual Piggy to access backend services.

**APIKey**- this property specifies the api key provided by Virtual Piggy to access backend services.

**trSrvcServerURL**- URL hosting Virtual Piggy service.

**oneStepPaymentMerchant**–specifies if the merchant is setup with Virtual Piggy to authorize and charge in one transaction (*true*) or authorize first and then charge at the time of settlement (*false*). Default is *false*.

**merchantCapturesPayment–**for a two-step merchant, this property specifies if the order payment settlement transactions are handled by the merchant (*true*) or by Virtual Piggy (*false*). Default is *false*.

# Purchase Process Configuration

### /com/virtualpiggy/integration/purchase/VirtualPiggyPurchaseProcessHelper

**successTrStatuses** – comma separated list of return statuses that indicate successful payment processing by Virtual Piggy. Default values are *Processed,ApprovalPending*.

Note: Any status value not in the above list is considered error and the order is not processed or submitted.

### /com/virtualpiggy/integration/purchase/VirtualPiggyShippingInfoDroplet

**defaultShippingMethod** – this property specifies the default shipping method that should be configured to be used with Virtual Piggy Child Checkout. Default is *Ground* used with CRS module.

### /com/virtualpiggy/integration/profile/VPProfileTools

**contactInfoToVPAddressMap** – this property maps the virtual piggy child/parent address to transient contact Info stored in ATG profile used during checkout as a shipping address.

# Callback Configuration

### /com/virtualpiggy/integration/callback/VirtualPiggyCallbackHandler

**successStatuses**– comma separated list of callback statuses that should be processed as successful status values. Default is *Processed*.

**rejectStatuses**- comma separated list of callback statuses that should be processed as rejected status values. Default is *Rejected*.

Note: Any status value that is not in one of the above lists is processed as Error.