Siebel ATG Implementation Guide

Order Integration

ORACLE

Implementation Guide for the integration of the ATG Order with the Siebel Quote.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.
This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

This documentation is in prerelease status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your beta trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Software License and Service Agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Table of Contents

Introduction

This Implementation Guide (IG) is provided as documentation for the Order Integration Reference Implementation (RI), The RI enables Oracle ATG Web Commerce to capture web orders in a format that provides all of the information required to convert from the ATG Order format to the Siebel Quote format.

This version of the document provides the necessary ATG order extensions required to support the Siebel Product Configurator integration (see reference 1). These extensions are mainly concerned with the structure of the order line items and pricing of the order.

This version supports a range of functionality from the addition of products to the cart, right up to the point where the order is ready to be checked out. Later versions of this integration will add support for the check-out process. Those extensions will focus on shipping and billing details.

The functionality in this version depends on the previous integrations for Account and Product Catalog.

Siebel has the concept of both quotes and orders. In this version of the integration ATG submits the shopping carts in Siebel quote format. Those quotes are converted into Siebel orders before they are sent to the downstream systems.

There are no changes required to Siebel for this integration. The integration uses existing Siebel web services.

Note: for the remainder of this document the terms Siebel quote and Siebel order may be used interchangeably but from a technical perspective we are referring to Siebel quotes.

References

[1] Siebel ATG Implementation Guide – Product Configurator Integration v1.0.

[2] Oracle ATG Web Commerce Programming Guide Version 10.2

[3] Oracle ATG Web Commerce Platform Programming Guide Version 10.2

[4] Oracle ATG Web Commerce Repository Guide Version 10.2

[5] Oracle ATG Web Commerce Commerce Reference Store Overview Version 10.2

Software Dependencies

This guide is based on the following software and respective versions:

- Oracle Siebel Version 8.1.1.10.
- Oracle ATG Web Commerce Version 10.2.

Technical Requirements Summary

This chapter gives a brief overview of the high level requirements for order capture with Oracle ATG Web Commerce.

Order Requirements

The ATG order structure must support the follow Siebel concepts:

- Siebel line items may be organised in a hierarchal tree where some line items may be parents of others. ATG's order structure must support this type of relationship between line items.
- Siebel products may have zero or more attribute / value pairs. ATG must support the persisting of multiple attributes and their values, against any line item in the order.

Pricing Requirements

The ATG order structure and pricing mechanisms must support the follow Siebel pricing concepts:

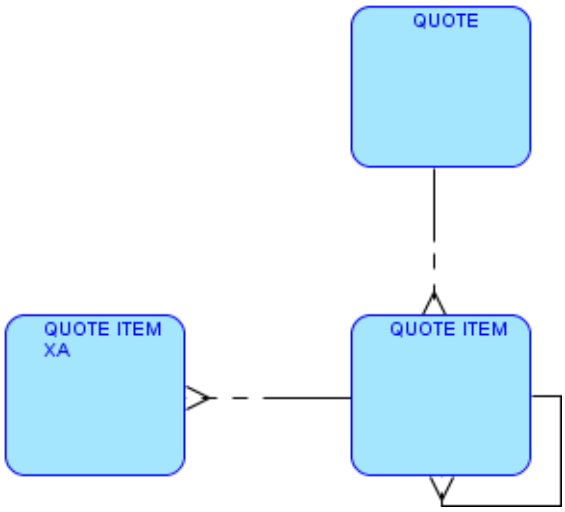
- Item pricing must be done by making web service calls to Siebel.
- Item pricing must support the storage of non-recurring and monthly recurring line item totals.

Order Repository Extensions

This chapter gives a brief overview of the extensions required to the ATG order repository to support conversion to the Siebel Quote format.

Siebel Quote

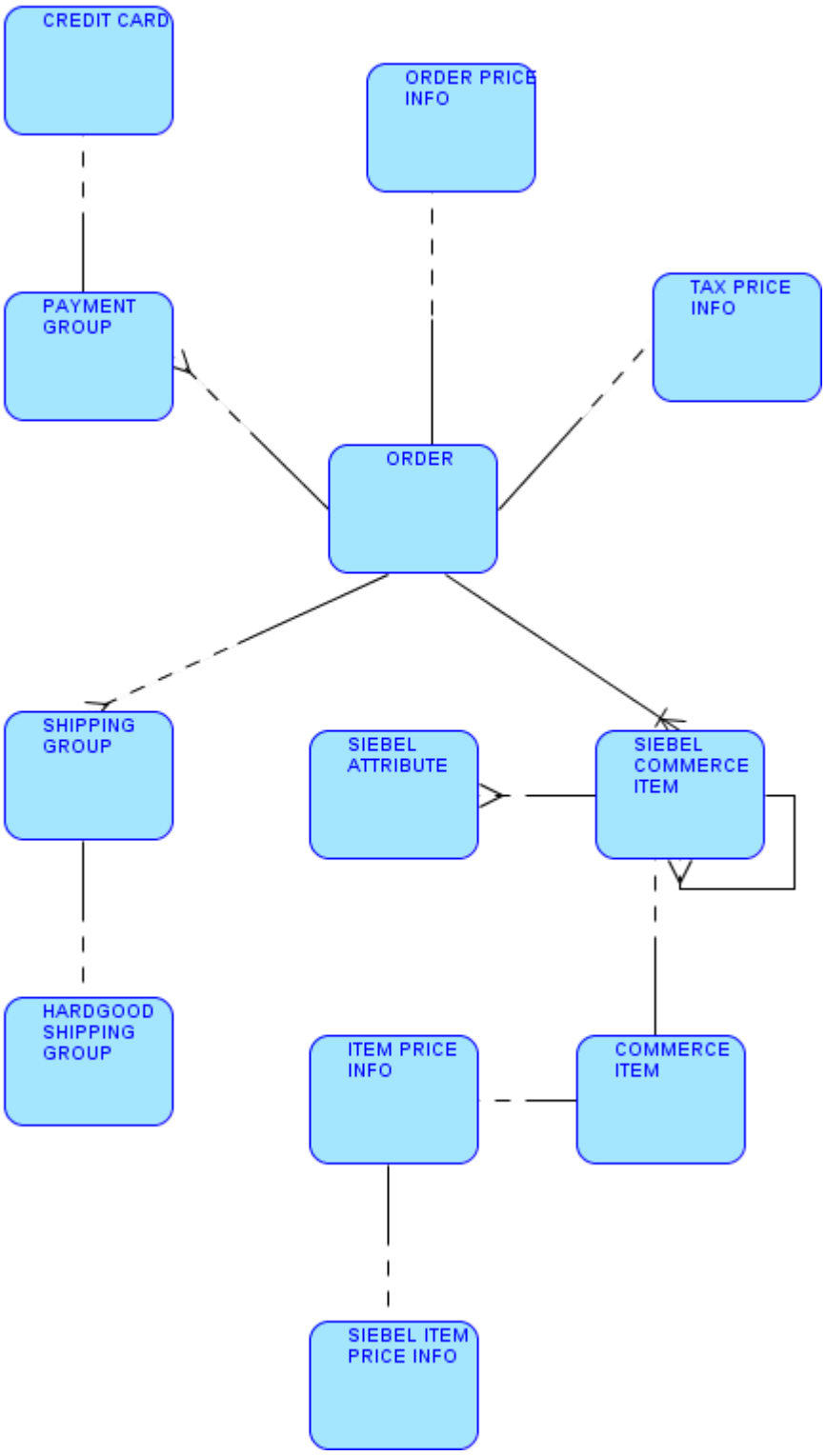
The diagram below gives a pictorial representation of the Siebel Quote entities and how they relate to each other.



To get a more detailed description of the Siebel entities see Appendix A.

ATG Order

The diagram below gives a pictorial representation of the ATG order entities (with the required Siebel extensions) and how they relate to each other.


CREDIT CARD
ORDER PRICE INFO
PAYMENT GROUP
TAX PRICE INFO
ORDER
SHIPPING GROUP
SIEBEL ATTRIBUTE
SIEBEL COMMERCE ITEM
HARDGOOD SHIPPING GROUP
ITEM PRICE INFO
COMMERCE ITEM
SIEBEL ITEM PRICE INFO

The main extensions to the ATG order repository are as follows:

- A new item descriptor known as Siebel Commerce Item has been added. This is a sub type of the existing Commerce Item. This extension also necessitates updating the Commerce Item definition to add the new sub type.
- A new Siebel Attribute item descriptor has been added to handle attributes attached to an order line item (i.e. a Commerce Item)
- A new Siebel Item Price Info item descriptor has been added to hold Siebel pricing information such as non-recurring and monthly recurring pricing information. This item descriptor is a sub type of the existing Item price Info item descriptor. The change also requires updating the Commerce Item item descriptor to have its priceInfo property be of type Siebel Item Price Info. It also necessitates an update to the Amount Info item descriptor to add a new type option for the Siebel Item Price Info item descriptor.

To get a more detailed description of the ATG repository elements and the extensions, see Appendix B.

ATG – Siebel Mappings

The basic mappings from the Siebel Quote format to the ATG order format are detailed in the table below:

Siebel Logical Entity	ATG Item Descriptor	ATG Item Descriptor Status
Quote	order	Existing item descriptor.
QuoteItem	siebelCommerceItem, siebelItemPriceInfo	Extension to existing commerceItem and the existing itemPriceInfo.
QuoteItemXA	siebelAttribute	New item descriptor.

For detailed mappings, see Appendix C.

Commerce Wrapper Extensions

This chapter specifies the extensions required for the commerce wrappers. Commerce wrappers are classes that wrap the order repository items.

Siebel QuoteItem

The Siebel QuoteItem corresponds to the ATG CommerceItem. These are essentially order line items.

Comparing Siebel QuoteItem to ATG CommerceItem

The main differences between an out of the box ATG commerce item and the corresponding Siebel quote item are:

1) The Siebel quote items may contain child quote items.

2) The Siebel quote items may have a collection of attributes.

3) The Siebel quote item has extra pricing information such as monthly and non-recurring charges.

To model these extensions two new wrapper classes are required; one for the commerce item extensions and one for the item price info extensions. The new classes are called SiebelCommerceItem and SiebelItemPriceInfo respectively. The next sections detail these new classes.

SiebelCommerceItem

This section details the new wrapper class that extends the commerce item.

Class Declaration

The SiebelCommerceItem class (/atg/siebel/order) is an extension of the vanilla CommerceItem wrapper class (/atg/commerce/order). The new class declaration is:

public class SiebelCommerceItem extends CommerceItemImpl implements CommerceItemContainer, SiebelAttributeContainer

As you can see from the class declaration the extension class also implements the existing CommerceItemContainer interface (/atg/commerce/order), as well as the new SiebelAttributeContainer interface (/atg/siebel/order). See Siebel QuoteItemXA section for details on that interface.

The CommerceItemContainer interface enables the SiebelCommerceItem class to serve as a container for a collection of child SiebelCommerceItem objects. This relates to the first comparison point in the section above.

The new SiebelAttributeContainer interface enables the SiebelCommerceItem class to act as a container for a collection of SiebelAttribute objects. This relates to the second comparison point in the section above.

Class Data

The class must maintain the following data:

- Action code (action to be taken in relation to the commerce item, i.e. no action, add, update or delete).
- Commerce item container (container for the child commerce items).
- Attribute container (container for the attributes).

Class General Implementation

The new class must also provide a method to handle the associated repository item.

Set Repository Item

public void setRepositoryItem (MutableRepositoryItem)

The method implementation must set the parent CommerceItem's repositoryItem and set the repositoryItem in the commerce item container.

CommerceItemContainer Implementation

The following methods must be implemented from the CommerceItemContainer interface:

Add Commerce Item

public void addCommerceItem (CommerceItem pCommerceItem) throws DuplicateCommerceItemException, InvalidParameterException;

Adds the commerce item to the commerce item container and to the repository property collection named commerceItems.

Add Commerce Item

public void addCommerceItem (CommerceItem pCommerceItem, int pIndex) throws DuplicateCommerceItemException, InvalidParameterException;

Adds the commerce item to the commerce item container at the given index. It also adds the commerce item to the repository property collection named commerceItems.

Remove Commerce Item

public CommerceItem removeCommerceItem (String pCommerceItemId) throws CommerceItemNotFoundException, InvalidParameterException;

Removes the commerce item from the commerce item container and from the repository property collection named commerceItems.

Remove All Commerce Items

public void removeAllCommerceItems ();

Removes all of the commerce item from the commerce item container and clears the repository property collection named commerceItems.

Get Commerce Item

public CommerceItem getCommerceItem (String pCommerceItemId) throws CommerceItemNotFoundException, InvalidParameterException;

Returns the commerce item from the container, matching on the commerce item ID.

Get Commerce Items By Catalog Ref Id

public List<SiebelCommerceItem> getCommerceItemsByCatalogRefId (String pCatalogRefId) throws CommerceItemNotFoundException, InvalidParameterException;

Returns the commerce items from the container, matching on the catalog reference ID.

Get Commerce Item Count

public int getCommerceItemCount ();

Gets the count of commerce items in the container.

Get Total Commerce Item Count

public long getTotalCommerceItemCount ();

Returns a count of the commerce items in the container but takes into account the quantity of items in each commerce item.

The implementation also adds the method:

Get Commerce Items

public List<SiebelCommerceItem> getCommerceItems()

Returns the list of commerce items from the container.

SiebelAttributeContainer Implementation

The following methods must be implemented from the SiebelAttributeContainer interface:

Get Siebel Attributes

public List getSiebelAttributes ();

Returns the attribute list from the container.

Add Siebel Attribute

public void addSiebelAttribute (SiebelAttribute pAttribute) throws DuplicateAttributeException, InvalidParameterException;

Adds the attribute to the attribute container and to the repository item collection property named attributes.

Add Siebel Attribute

public void addSiebelAttribute (SiebelAttribute pAttribute, int pIndex) throws DuplicateAttributeException, InvalidParameterException;

Adds the attribute to the attribute container at the specified index and also to the repository item collection property named attributes.

Remove Attribute

public SiebelAttribute removeAttribute (String pAttributeId) throws AttributeNotFoundException, InvalidParameterException;

Removes the attribute from the attribute container and from the repository item collection, attributes.

Remove All Attributes

public void removeAllAttributes();

Removes all attributes from the attribute container and from the repository item collection property, attributes.

Get Attribute

public SiebelAttribute getAttribute (String pAttributeId) throws AttributeNotFoundException, InvalidParameterException;

Gets the attribute specified by the attribute id, from the container.

Get Attribute Count

public int getAttributeCount();

Returns the number of attributes in the attribute container.

SiebelItemPriceInfo

This section details the new wrapper class that extends the item price info wrapper.

Class Declaration

The SiebelItemPriceInfo class (/atg/siebel/pricing) is an extension of the ootb ItemPriceInfo wrapper class (/atg/commerce/pricing). The new class declaration is:

public class SiebelItemPriceInfo extends ItemPriceInfo

Class Data

The class must maintain the following data:

- Monthly recurring charges.
- Non-recurring charges.

Class General Implementation

The class provides getter and setter methods for the two properties.

Siebel QuoteItemXA

The Siebel QuoteItemXA requires a new wrapper class on ATG. The new class is known as SiebelAtribute. The new class uses a SiebelAttributeConatiner object in order to store the attributes.

SiebelAttribute

This section details the new wrapper class.

Class Declaration

The SiebelAttribute class (/atg/siebel/order) is a wrapper class for a Siebel attribute. The new class declaration is:

public class SiebelAttribute extends CommerceIdentifierImpl implements ChangedProperties

This new wrapper class extends the existing CommerceIdentifierImpl class (/atg/commerce/order) and it implements the existing ChangedProperties interface (/atg/commerce/order).

Class Data

The class must maintain the following data:

- Repository Item (the repository item representing the attribute).
- Save All Properties Flag (indicates whether or not all properties need to be saved for the attribute).
- Changed Flag (indicates whether or not any properties have been changed since the last save).
- Changed Properties (a set of properties that have changed since the last save).

- This class must also provide getter and setter methods for each of the repository item's properties. The properties for the SiebelAttribute class are: attributeDefinitionId, attributeName, actionCode and value. The getter and setter methods get and set the values from the repository item by calling getPropertyValue and setPropertyValue.

ChangedProperties Implementation

The ChangedProperties interface provides methods for determining what properties need to be saved to the repository. The following methods must be implemented:

Update

public void update (Observable pChangedObject, Object pPropertyName)

Called whenever a property has been changed.

Get Save All Properties

public boolean getSaveAllProperties()

Returns the value for the Save All Properties flag.

Set Save All Properties

public void setSaveAllProperties(boolean pSaveAllProperties)

Sets the value for the Save All Properties flag.

Is Changed

public boolean isChanged()

Returns true if the Changed flag is set or, if the Changed Properties set has some entries.

Set Changed

public void setChanged (boolean pChanged)

Sets the Changed flag.

Get Changed Properties

public Set getChangedProperties()

Returns the Changed Properties set.

Add Changed Property

public void addChangedProperty(String pPropertyName)

Adds a property name to the Changed Properties set.

Clear Changed Properties

public void clearChangedProperties()

Clears the set of Changed Properties.

Get Property Value

public Object getPropertyValue (String pPropertyName)

Gets a property value from the repository item.

Set Property Value

public void setPropertyValue (String pPropertyName, Object pPropertyValue)

Sets a property value on the repository item.

SiebelAttributeContainer Interface

This section details the new SiebelAttributeContainer interface:

Get Siebel Attributes

public List getSiebelAttributes();

Returns a List of attributes.

Add Siebel Attribute

public void addSiebelAttribute (SiebelAttribute pAttribute) throws DuplicateAttributeException, InvalidParameterException;

Add an attribute to the container. If the attribute was already in the container a DuplicateAttributeException is thrown.

Add Siebel Attribute

public void addSiebelAttribute (SiebelAttribute pAttribute, int pIndex) throws DuplicateAttributeException, InvalidParameterException;

Add an attribute to the container at the given index. If the attribute was already in the container a DuplicateAttributeException is thrown.

Remove Attribute

public SiebelAttribute removeAttribute (String pAttributeId) throws AttributeNotFoundException, InvalidParameterException;

Removes the attribute whose id is passed as a parameter. If the attribute is not in the container then a AttributeNotFoundException is thrown. Otherwise a reference to the removed attribute is returned.

Remove All Attributes

public void removeAllAttributes();

Removes all of the attributes from the container.

Get Attribute

public SiebelAttribute getAttribute (String pAttributeId) throws AttributeNotFoundException, InvalidParameterException;

Returns the attribute specified by the id passed in. If the attribute with the corresponding id is not in the container then an AttributeNotFoundException is thrown.

Get Attribute Count

public int getAttributeCount();

Returns the number of attributes in the container.

SiebelAttributeContainer Implementation

This section details the new wrapper class.

Class Declaration

The SiebelAttributeContainerImpl class is an implementation of the SiebelAttributeContainer interface. The class declaration is:

public class SiebelAttributeContainerImpl implements SiebelAttributeContainer

Class Data

The class must maintain the following data:

- Siebel Attribute list.

Class Implementation

The implementation follows the interface descriptions and works on the Siebel Attribute list detailed above.

Commerce Pipeline Extensions

This chapter describes the extensions to the commerce pipeline required to support Siebel quotes.

Pipeline Processors

There are two new pipeline processors required to support the Siebel quote / order integration. These processors are concerned with the loading and saving of attributes.

Update Order Processor

This processor is implemented in a new class named, ProcSaveAttributeObjects (atg/Siebel/order/processor).

Class Declaration

The ProcSaveAttributeObjects class (/atg/siebel/order/processor) is an implementation of the PipelineProcessor interface (/atg/service/pipeline). The class declaration is:

public class ProcSaveAttributeObjects extends SavedProperties implements PipelineProcessor

Class Implementation

The implementation has two methods as follows:

Run Process

public int runProcess (Object pParam, PipelineResult pResult) throws Exception

This method is the point of entry for the processor's processing. It is called from the ATG pipeline mechanism (see references 2 & 3 for more details on commerce pipelines).

The first step is to extract the order, order manager and order repository references from the parameter map.

The next step is to get a handle on order tools and to get a mutable version of the order repository.

Finally the method calls saveAttributes to do the actual work. It passes the order, the list of order commerce items, a reference to order manager, a reference to order tools and a reference to the mutable repository, as parameters.

Save Attributes

protected boolean saveAttributes (Order pOrder, List<SiebelCommerceItem> pSiebelCommerceItems, OrderManager pOrderManager, OrderTools pOrderTools, MutableRepository pMutableOrderRepository) throws Exception

This method does the work of storing the attributes. It processes each of the commerce items passed in the list.

For each commerce item the method calls getSiebelAttributes to get the list of attributes for the commerce item.

For each attribute the method does the following:

- Establishes the list of properties to be saved on the attribute.
- Get the repository item from the attribute object and if it is not there get it (for update) from the mutable repository.
- Navigate the list of properties to be saved and for each, ensure that the property exists and save its value.
- If the attribute is transient then persist it.
- Update the repository item in the order repository.
- Set the changed properties flag on the SiebelAttribute class.

When all attributes have been processed on a commerce item, check if that commerce item has children. If it has, process each of those children recursively, so that all attributes associated with any commerce item in the commerce item tree, are saved.

Refresh Order Processor

This processor is implemented in a new class named, ProcLoadAttributeObjects (atg/siebel/order/processor).

Class Declaration

The ProcLoadAttributeObjects class is an implementation of the PipelineProcessor interface (/atg/service/pipeline). The class declaration is:

public class ProcLoadAttributeObjects extends LoadProperties implements PipelineProcessor

Class Implementation

The implementation has two methods as follows:

Run Process

public int runProcess (Object pParam, PipelineResult pResult) throws Exception

This method is the point of entry for the processor's processing. It is called from the ATG pipeline mechanism (see references 2 & 3 for more details on commerce pipelines).

The first step is to extract the order, order manager, order repository references from the parameter map. Also from the parameter map the method gets the invalidate cache flag. Next it gets a reference to order tools.

Finally the method calls loadAttributes and passes the parameters, order, commerce item container (i.e. the order), order manager, order tools and the invalidate cache flag.

Load Attributes

protected void loadAttributes (Order pOrder, CommerceItemContainer pCommerceItemContainer, OrderManager pOrderManager, OrderTools pOrderTools, Boolean pInvalidateCache) throws Exception

This method does the work of loading the commerce item attributes. It processes each of the commerce items passed in the container.

For each commerce item the method gets the attribute items from the commerce item's repository item.

For each attribute item the method does the following:

- Check if the cache must be invalidated for the item. If so call invalidateCache.
- Create a new instance of the class for the attribute.
- Set the id on the class object to the attribute's repository item id.
- Set the repository item on the class object.
- Load each of the properties and set them on the class object.
- Add the attribute object to the commerce item.

Once all attributes have been loaded for the commerce item, this method checks if the commerce item has any children. If it has then this method will recursively call loadAttributes for each child. This ensures that all attributes right down through the commerce item tree are loaded.

Commerce Pipeline Definition

The following XML needs to be added to the commercepipeline.xml file (/atg/commerce):

```
<pipelinemanager>

  <pipelinechain name="updateOrder" transaction="TX_REQUIRED"
headlink="updateOrderObject" xml-combine="append">
    <pipelinelink name="updateCommerceItemObjects" transaction="TX_MANDATORY" xml-
combine="replace">
      <processor jndi="/atg/commerce/order/processor/SaveCommerceItemObjects"/>
      <transition returnvalue="1" link="updateAttributeObjects"/>
    </pipelinelink>
```

```
    <pipelinelink name="updateAttributeObjects" transaction="TX_MANDATORY">
      <processor jndi="/atg/siebel/order/processor/SaveAttributeObjects"/>
      <transition returnvalue="1" link="updateShippingGroupObjects"/>
    </pipelinelink>
  </pipelinechain>

  <pipelinechain name="refreshOrder" transaction="TX_REQUIRED"
headlink="loadOrderObjectForRefresh" xml-combine="append">
    <pipelinelink name="loadCommerceItemObjects" transaction="TX_MANDATORY" xml-
combine="replace">
      <processor jndi="/atg/commerce/order/processor/LoadCommerceItemObjects"/>
      <transition returnvalue="1" link="loadAttributeObjects"/>
    </pipelinelink>
    <pipelinelink name="loadAttributeObjects" transaction="TX_MANDATORY">
      <processor jndi="/atg/siebel/order/processor/LoadAttributeObjects"/>
      <transition returnvalue="1" link="loadShippingGroupObjects"/>
    </pipelinelink>
  </pipelinechain>

</pipelinemanager>
```

This inserts the two new attribute processors into the appropriate pipeline chains.

Purchase Process Extensions

This chapter details the extensions required to the vanilla ATG purchase process classes, required to support the Siebel quote / order integration.

The first extension is to the cart modifier form handler and the second is to the purchase process helper class. These are used when adding products to the ATG shopping cart.

Cart Modifier Form Handler

This section details the extensions required to the CartModifierFormHandler class (/atg/commerce/order/purchase). The extensions are made in a class named SiebelCartModifierFormHandler (/atg/siebel/order/processor) that extends the CartModifierFormHandler class. The following method over-rides are required.

Merge Item Input For Add

In the ootb form handler this method (mergeItemInputForAdd) merges any commerce items that refer to the same product. In Siebel it is perfectly valid to have the same product in two separate line items and in many cases the order structure depends on it. In the over-ride, the call to the method mergeValueDictionaries is removed so that the items are not merged.

Do Add Items To Order

In this method (doAddItemsToOrder) extension the commerce item id is appended to the end of the addItemToOrderSuccessURL. The string “?commerceItemId=<id>” is appended to the URL, where <id > is replaced by the actual id of the newly added commerce item. This extension enables the commerce item id (of the newly added item) to be passed back to the UI.

Handle Edit Configurable Product

public boolean handleEditConfigurableProduct (DynamoHttpServletRequest pRequest, DynamoHttpServletResponse pResponse) throws ServletException, IOException

This method supports the editing of previously configured products. The page coder hooks the jsp up to this method from an ‘Edit’ button.

The method requires that the form handler have a reference to the configurator manager (see reference 1 for details on configurator manager).

The method retrieves the root product configuration instance by calling the createProductInstanceFromCommerceItem method on configurator manager, passing the commerce item id as a parameter.

Next it calls the performProductConfigurationEdit method on configurator manager, passing the root product configuration instance as a parameter. That calls sets up a Siebel configurator session in edit mode.

If the performProductConfigurationEdit method call was successful then the method does a re-direct to the jsp that enables the shopper to edit the product. In the configurator reference implementation the example shipped is named stand-alone-product.jsp (siebel.war/configurator/renderer). See the Standalone Product Configuration Initialisation Process section in reference 1 for details)

Purchase Process Helper

This section details the extensions required to the PurchaseProcessHelper class (/atg/commerce/order/purchase). The extensions are made in a class named SiebelPurchaseProcessHelper (/atg/siebel/order/processor) that extends the PurchaseProcessHelper class.

The first extension is to add a reference to the attribute manager (/atg/siebel/order/SiebelAttributeManager). This is needed when adding items to the order.

The remaining extensions are related to the addItemsToOrder class method.

Add Items To Order

This method has several signatures. The one to be over-ridden has the following signature:

public List addItemsToOrder (Order pOrder, ShippingGroup pShippingGroup, RepositoryItem pProfile, AddCommerceItemInfo[] pItemInfos, Locale pUserLocale, String pCatalogKey, PricingModelHolder UserPricingModels, PipelineErrorHandler pErrorHandler, Map pExtraParameters) throws CommerceException

The over-ridden method carries out the following steps:

- Check that a transaction exists.
- Process the item infos and for each with a quantity greater than zero do the following:
    - Call processSiebelProduct and pass the item info and catalog key.
    - Process each of the commerce items returned from the processSiebelProduct call.
        - Call commerce item manager to add the commerce item to the order.
        - Call the parent class methods, getShippingGroupForItem, addItemToShippingGroup, and processGiftAddition to add the shipping group and process gift additions.
        - Add the commerce item to a list that this method returns to the caller.
- Reprice the order by calling the runProcessRepriceOrder method. This results in the item pricing engine being called and as you can see from the Item Pricing Engine chapter below that this calls out to Siebel to do the item pricing.
- Generate scenario events for each of the commerce items added by calling runProcessSendScenarioEvent.

The over-ridden method above calls a new method named processSiebelProduct.

Process Siebel Product

This new method has the following signature:

private ArrayList<SiebelCommerceItem> processSiebelProduct (SiebelAddCommerceItemInfo pItemInfo, String pCatalogKey) throws CommerceException

This method processes the Siebel product by adding the appropriate commerce items to the order (depending on the product type). It also takes care of adding the corresponding attributes. It carries out the following processing:

- Get references to order manager, order tools and catalog tools.
- Call the getProductDetails method on catalog tools to get the product definition and structure.
- Get the product attributes from the object returned by the getProductDetails method.
- Initialise a list of commerce items to be returned to the caller.
- The next step depends on the product type:
    - Simple Product: call createSiebelCommerceItem (see below for details) and add the resulting commerce item to the list.
    - Simple Product with Attributes: call createSiebelCommerceItem for each unit in the quantity and for each add the resulting commerce item to the list. Pass the product attributes into the createSiebelCommerceItem call.
    - Simple Product Bundle: call createSiebelCommerceItem for each unit in the quantity and for each add the resulting commerce item to the list. Pass the product attributes into the createSiebelCommerceItem call. Get the child products from the catalog structure. For each child product: create an instance of SiebelAddCommerceItemInfo and fill in the values; get the attributes for the child and then call createSiebelCommerceItem; add the resulting commerce item to the list.
    - Configurable Product: call createSiebelCommerceItem for each unit in the quantity and for each add the resulting commerce item to the list. Pass the product attributes into the createSiebelCommerceItem call.
    - Promotion: call createSiebelCommerceItem for each unit in the quantity and for each add the resulting commerce item to the list.
- Return the list of commerce items created during the method execution.

The processSiebelProduct method above calls a method named createSiebelCommerceItem and here are the details for that method.

Create Siebel Commerce Item

The method signature is:

public SiebelCommerceItem createSiebelCommerceItem (SiebelAddCommerceItemInfo pItemInfo, ArrayList<SiebelCatalogAttribute> pCatalogAttributes, String pCatalogKey, long pQuantity) throws CommerceException

The method starts by getting references to the commerce item manager and the attribute manager. Next it uses the quantity parameter to set the value of quantity on the item info object, and calls createCommerceItem (see below for details).

Next the method calls the parent method setCommerceItemProperties to set the properties (from the item info) on the commerce item. For each attribute passed in the list we call createSiebelAttribute on the attribute manager to create the attribute with the default value; then we call addAttributeToItem on commerce item manager to add the attribute to the commerce item. Finally the commerce item is returned to the caller.

The final method relating to the addItemsToOrder over-rides is createCommerceItem.

Create Commerce Item

The signature is:

protected SiebelCommerceItem createCommerceItem (SiebelAddCommerceItemInfo pItemInfo, String pCatalogKey) throws CommerceException

This method uses commerce item manager to create a Siebel commerce item.

Finally the class has an utility method used to add a Siebel commerce item to the order.

Add Commerce Item To Order

The signature is:

protected SiebelCommerceItem addCommerceItemToOrder (Order pOrder, SiebelCommerceItem pCommerceItem) throws CommerceException

This method just uses commerce item manager to add the item to the order.

This extension class also creates two new classes named, SiebelAddCommerceItemInfo and SiebelAddCommerceItemAttribute. These are described below.

SiebelAddCommerceItemInfo

This class is used when adding a Siebel commerce item.

Class Declaration

The SiebelAddCommerceItemInfo class (atg/siebel/order/purchase) is an extension of AddCommerceItemInfo. The class declaration is:

public class SiebelAddCommerceItemInfo extends AddCommerceItemInfo

Class Data

The class adds the following data to the AddCommerceItemInfo class:

- An array of SiebelAddCommerceItemAttribute.

- An attribute count.

Class Implementation

The class provides the following methods:

Get Attributes

public SiebelAddCommerceItemAttribute [] getAttributes()

This method returns the list of SiebelAddCommerceItemAttribute objects.

Set Add Attribute Count

public void setAddAttributeCount (int pAddAttributeCount)

This method sets the number of attributes to allocate in the in the attributes array and creates a SiebelAddCommerceItemAttribute object for each.

SiebelAddCommerceItemInfo

This class is used when adding an attribute to a Siebel commerce item.

It is a simple bean class with three attributes; one for the attribute definition id, one for the action code and one for the value.

Catalog Tools Extensions

This chapter details the extensions required to ATG catalog tools. Only the SiebelCatalogTools extension class is documented below as the other related classes are all bean classes and can be easily understood by examining the source.

Siebel Catalog Tools Class

The SiebelCatalogTools class (atg.siebel.catalog) is a new class that implements the functionality required for the Siebel integration. It extends the CustomCatalogTools class (atg.commerce.catalog.custom).

Class Declaration

The class declaration is:

public class SiebelCatalogTools extends CustomCatalogTools

Class Data

The class maintains the following data:

- Item Descriptor Names (for the following item descriptors: category product, attribute and product class).
- Property Names (for the following properties: class id, id, inclusive eligibility flag, start date, end date, display name, price list id, product line id, product def type code, product type code, siebel type, price type, configured attributes, attribute definition id, attribute name, default value, required flag, read only flag, hidden flag, promotion structure, simple bundle structure, configurable product structure, relationships, product id, class name, minimum cardinality, maximum cardinality, default cardinality, aggregate default products, relationship domains, default product id, Siebel id, configured attributes, display value, values, attribute value, domain type, data type, product lines, type, and product line id.

Class Implementation

The class has the following methods:

Get Product Details

public SiebelCatalogProduct getProductDetails (String pProductId) throws RepositoryException, PropertyNotFoundException

This method returns details about the specified product. The details include the product properties, any associated attributes and references to any child products. The details are returned as a SiebelProductCatalog object (atg.siebel.catalog).

The method starts by calling the getBaseProductDetails to get the base details about the product. The base details include the product properties and any associated attributes.

Any additional information required depends on the product type. For simple products and simple products with attributes there are no additional details required. For, simple product bundles, configurable products and promotions we need the corresponding structures.

The structure for a promotion is added by calling the addPromotionChildRelationships method. The structure for a configurable product is added by calling the method, addConfigurableProductChildRelationships, and for a simple product bundle the method, addSimpleBundleChildRelationships is called.

Get Base Product Details

public SiebelCatalogProduct getBaseProductDetails (String pProductId) throws RepositoryException, PropertyNotFoundException

This method gets the product properties and any associated attributes.

The method starts by calling the getProductDefinition method to get the product definition repository item. It then creates the SiebelCatalogProduct object and sets the repository item on that object.

It then calls the method setProductProperties to extract the properties from the repository item and set them on the SiebelCatalogProduct object.

Next it calls the method addProductAttributes to extract the attributes and add them to the SiebelCatalogProduct object. That object is then returned to the caller.

Get Product Definition

protected RepositoryItem getProductDefinition (String pProductId) throws RepositoryException

This method retrieves the product definition from the product catalog repository.

Set Product Properties

private void setProductProperties (SiebelCatalogProduct pProductDetails) throws PropertyNotFoundException

This method uses ATG's DynamicBeans to extract the property values from the product repository item, and sets these values on the SiebelProductCatalog object.

Add Product Attributes

private void addProductAttributes (SiebelCatalogProduct pProductDetails) throws PropertyNotFoundException, RepositoryException

This method extracts the attributes from the product definition (i.e. configured attributes) and uses those to set the attributes (i.e. atg.siebel.catalog.SiebelCatalogAttribute) on the SiebelProductCatalog object. Not all property values come from the configured attribute though; quite a few come from the corresponding base attribute.

The method starts by retrieving the configured attributes from the product definition and iterating through them.

For each configured attribute the method extracts the attribute definition id and uses that to get the corresponding base attribute by calling the getBaseAttribute method. The values from the base are used to set the corresponding property values on the attribute, including any enumerated values for attribute selection.

The next step is to extract the values from the configured attribute so that we can set the remaining properties on the SiebelCatalogAttribute object. Some of these values will over-ride those set from the base attribute, but only if a value is actually specified for that property.

Get Base Attribute

private RepositoryItem getBaseAttribute(String pAttributeDefinitionId) throws RepositoryException

This method gets the base attribute repository item from the product catalog repository, using the attribute definition id as the key.

Add Promotion Child Relationships

private void addPromotionChildRelationships (SiebelCatalogProduct pProductDetails) throws RepositoryException, PropertyNotFoundException

This method extracts the promotion's root child product relationships from the promotion structure definition and adds them to the SiebelCatalogProduct object.

The method starts by retrieving the product definition repository item and extracting the promotion structure item from that. Next it extracts the relationships from the promotion structure.

For each relationship in the promotion structure we create a new SiebelCatalogRelationship object. We set some basic properties on the object before figuring out what type of relationship it is.

The relationship type is established by examining the type property.

If the type is an 'Aggregate' relationship then we must check what defines the relationship, i.e. either product line or product class.

First we check if the productLineId has been populated and if it has we call the method getProductLineProductIds to retrieve the list of products belonging to that product line. If that is not populated then we check the className to see if it is populated. If it is then we call the getProductClassProductIds method to retrieve the list of products belonging to that product class. We call the getProductDetails for each product and add them as child products (to the relationship).

If the relationship type is 'Components' then we extract the product id from the relationship and call getProductDetails to get the details for the child product. The child product is then added to the relationship.

When all relationships have been processed we will have created the entire product tree for the promotion. The relationships are set on the SiebelCatalogProduct object by calling its setChildRelationships method.

Add Configurable Product Child Relationships

private void addConfigurableProductChildRelationships (SiebelCatalogProduct pProductDetails) throws RepositoryException, PropertyNotFoundException

This method extracts the configurable product's relationships from the configurable product structure definition and adds them to the SiebelCatalogProduct object.

The method starts by retrieving the product definition repository item and extracting the configurable product structure item from that. Next it extracts the relationships from the configurable product structure.

For each relationship in the configurable product structure we create a new SiebelCatalogRelationship object. We set some basic properties on the object before extracting the relationship domains from the relationship. For each relationship domain we get the siebel id property and call the getProductDetails method to recursively get the child product details and its product tree. The child products are added to the SiebelCatalogRelationship object.

Finally we call the setChildRelationships method on the SiebelCatalogProduct object.

Add Simple Bundle Child Relationships

private void addSimpleBundleChildRelationships (SiebelCatalogProduct pProductDetails) throws RepositoryException, PropertyNotFoundException

This method extracts the simple bundle's relationships from the simple bundle structure definition and adds them to the SiebelCatalogProduct object.

The method starts by retrieving the product definition repository item and extracting the simple bundle structure item from that. Next it extracts the relationships from the simple bundle structure.

For each relationship in the simple bundle structure we create a new SiebelCatalogRelationship object. We set some basic properties on the object before extracting the relationship domains from the relationship. For each relationship domain we get the siebel id property and call the getProductDetails method to recursively get the child product details and its product tree. The child products are added to the SiebelCatalogRelationship object.

Finally we call the setChildRelationships method on the SiebelCatalogProduct object.

Get Navigable Products

public Set<SiebelCatalogProduct> getNavigableProducts () throws RepositoryException, PropertyNotFoundException

This method returns the details for a list of products that are accessible through the catalog navigation hierarchy, i.e. the products referenced by catalog product objects.

The method queries for promotions and configurable products and gets the details for each.

Get Products For Product Line

public ArrayList<SiebelCatalogProduct> getProductsForProductLine (String pProductLineId) throws RepositoryException, PropertyNotFoundException

This method gets the base details for the products contained in the product line specified by the input parameter. It is used to get the child products for a promotion aggregate relationship. The method doesn't get any structure associated with these child products as the details are only used to display the immediate child products (of the promotion relationship) in the promotion edit process.

The method calls the getProductLineProductIds method to get the product ids and then it calls the getBaseProductDetails method for each.

Get Products For Product Class

public ArrayList<SiebelCatalogProduct> getProductsForProductClass (String pClassName) throws RepositoryException, PropertyNotFoundException

This method gets the base details for the products contained in the product class specified by the input parameter. It is used to get the child products for a promotion aggregate relationship. The method doesn't get any structure associated with these child products as the details are only used to display the immediate child products (of the promotion relationship) in the promotion edit process.

The method calls the getProductClassProductIds method to get the product ids and then it calls the getBaseProductDetails method for each.

Get Product Line Product Ids

private ArrayList<String> getProductLineProductIds (String pProductLineId) throws RepositoryException

The method retrieves a list of product ids; one for each product belonging to the product line specified by the input parameter. It checks the each product's collection property, productLines, to establish which products reference the given product line.

Get Product Class Product Ids

private ArrayList<String> getProductClassProductIds (String pClassName) throws RepositoryException

The method retrieves a list of product ids; one for each product belonging to the product class specified by the input parameter.

The first step is to query the product class item descriptor to find the class id that corresponds to the class name, given in the parameter.

The method then checks the each product's classId property to establish which products reference the given product class.

Catalog Tools Component

The existing CatalogTools component (atg.commerce.catalog.CatalogTools.properties) is redefined to change the class and add the new properties. The new component definition has the following entries:

$class=atg.siebel.catalog.SiebelCatalogTools

categoryProductItemDescriptor=category-product

attributeItemDescriptor=attribute

productClassItemDescriptor=product-class

classIdPropertyName=classId

idPropertyName=id

inclusiveEligibilityFlagPropertyName=inclusiveEligibilityFlag

effectiveFromPropertyName=startDate

effectiveToPropertyName=endDate

namePropertyName=displayName

productClassNamePropertyName=name

priceListIdPropertyName=priceListId

productLineIdPropertyName=productLineId

productDefTypeCodePropertyName=productDefTypeCode

productTypeCodePropertyName=productTypeCode

productTypePropertyName=siebelType

priceTypePropertyName=priceType

attributesPropertyName=configuredAttributes

attributeDefinitionIdPropertyName=attributeDefinitionId

attributeNamePropertyName=attributeName

defaultValuePropertyName=defaultValue

requiredFlagPropertyName=requiredFlag

readOnlyFlagPropertyName=readOnlyFlag

hiddenFlagPropertyName=hiddenFlag

promotionStructurePropertyName=promotionStructure

simpleBundleStructurePropertyName=simpleBundleStructure

configurableProductStructurePropertyName=configurableProductStructure

relationshipsPropertyName=relationships

#productIdPropertyName=productId

catProdProductIdPropertyName=productId

classNamePropertyName=className

minimumCardinalityPropertyName=minimumCardinality

maximumCardinalityPropertyName=maximumCardinality

defaultCardinalityPropertyName=defaultCardinality

aggregateDefaultProducts=aggregateDefaultProducts

relationshipDomains=relationshipDomains

defaultProductIdPropertyName=defaultProductId

siebelIdPropertyName=siebelId

configuredAttributesPropertyName=configuredAttributes

displayValuePropertyName=displayValue

attributeValuesPropertyName=values

attributeValuePropertyName=attributeValue

attributeDomainTypePropertyName=domainType

attributeDataTypePropertyName=dataType

productLinesPropertyName=productLines

relationshipTypePropertyName=type

productLineIdPropertyName=productLineId

Order Manager Extensions

This chapter details the required extensions to the order manager.

OrderManager Component

The OrderManager component (/atg/commerce/order/OrderManager.properties) is redefined to change the value of the addItemInfoClass property. The new component definition has the following entry:

addItemInfoClass=atg.siebel.order.purchase.SiebelAddCommerceItemInfo

Order Tools Extensions

This chapter details the extensions made to the ATG OrderTools class, to support the Siebel quote. The extension class is named, SiebelOrderTools.

SiebelOrderTools Class

This section details the new SiebelOrderTools class.

Class Declaration

The SiebelOrderTools class (/atg/siebel/order) is an extension of the ootb OrderTools class (/atg/commerce/order). The new class declaration is:

public class SiebelOrderTools extends OrderTools

Class Data

The class must maintain the following data:

- A property that references the CommerceItemManager component.
- A property that references the SiebelAttributeManager component.
- A property that references the PriceListManager component.
- A property holding the name of the siebel attribute item descriptor and one for each of the following repository property names: the user siebel account id, the organization parent organization, organization name, the organization siebel id, the price list id, the price list and the default price list.

Class Implementation

This methods in this class generally handle adding, updating and deleting items from the order whilst, at the same time, handling the conversion between the Siebel quote format and the ATG order format.

The following methods are implemented in the class:

Add Quote Item To Order

This method has the following signature:

public SiebelCommerceItem addQuoteItemToOrder(Order pOrder, String pParentItemId, QuoteItem pQuoteItem, String pCatalogKey) throws TransformException, CommerceException

This method converts a Siebel QuoteItem to a SiebelCommerceItem (or a tree of SiebelCommerceItems) and adds the commerce item tree to the order. If a parent item is specified, the commerce item tree is added to that instead of at the order level.

Update Quote Item In Order

public SiebelCommerceItem updateQuoteItemInOrder(Order pOrder, String pParentItemId, String pItemId, QuoteItem pQuoteItem, String pCatalogKey) throws TransformException, CommerceException

This method converts a Siebel QuoteItem to a tree of SiebelCommerceItems and uses that to replace the given commerce item (tree) in the order. If the parent item is specified then the tree is replaced under that instead of at the order level. The update is achieved by first removing the old item and then adding the new item with the same id as the old.

Add Product Config Instance To Order

public SiebelCommerceItem addProductConfigInstanceToOrder (Order pOrder, RootProductConfigInstance pRootInstance, String pCatalogKey) throws TransformException, CommerceException

This method converts a root product configuration instance to a SiebelCommerceItem (or a tree of SiebelCommerceItems), and adds the commerce item tree to the order. If a parent item is specified, the commerce item tree is added to that instead of at the order level.

Update Product Config Instance In Order

public SiebelCommerceItem updateProductConfigInstanceInOrder (Order pOrder, String pExistingItemId, RootProductConfigInstance pRootInstance, String pCatalogKey) throws TransformException, CommerceException

This method converts a root product configuration instance to a SiebelCommerceItem (or a tree of SiebelCommerceItems) and uses that to replace the given commerce item (tree) in the order. If the parent item is specified then the tree is replaced under that instead of at the order level. The update is achieved by first removing the old item and then adding the new item with the same id as the old.

Delete Item From Order

public void deleteItemFromOrder(Order pOrder, String pParentItemId, String pItemId) throws TransformException, CommerceException

This method deletes the specified commerce item (tree). If the parent is not null then the method searches for the item under that, otherwise it searches at the order level.

Convert Order To Siebel Quote

protected Quote convertOrderToSiebelQuote (Order pOrder) throws TransformException

This method takes and ATG order and converts it into a Siebel quote. It starts by extracting the profile from the order and using that to find the related Siebel account information. Next it calls createSiebelQuote to create the basic Siebel quote information and finally it calls convertCommerceItemsToQuoteItems to convert the commerce items into their equivalent Siebel quote items.

Create Siebel Quote

protected Quote createSiebelQuote (Order pOrder, String pAccountName, String pAccountId, String pPriceListId) throws CommerceException

This method creates the Siebel quote object and populates the header fields. For this release header fields consists of the account related fields, the quote id and the price list id.

Convert Commerce Items To Quote Items

protected List<QuoteItem> convertCommerceItemsToQuoteItems (List<SiebelCommerceItem> pCommerceItems, SiebelCommerceItem pParentCommerceItem, String pAccountId) throws RepositoryException, PropertyNotFoundException

This method converts ATG commerce items into Siebel Quote items. For each ATG commerce item, it calls createSiebelQuoteItem to produce the corresponding Siebel quote item. If the commerce item has child items then they must be also processed to create the proper quote item hierarchy.

When processing the commerce item children to produce the quote item hierarchy we must first look at the product type. For promotions the root child products must be added at the same level as the parent promotion. For all other products the children are added underneath the parent as per normal.

Create Siebel Quote Item

protected QuoteItem createSiebelQuoteItem(SiebelCommerceItem pCommerceItem, SiebelCommerceItem pParentCommerceItem, String pAccountId) throws RepositoryException, PropertyNotFoundException

This method creates a Siebel quote item with the corresponding quote item XA items (i.e. attributes) from the ATG commerce item.

It sets all of the quote item fields as per the mappings set out in Appendix C. It also calls createListQuoteItemXA to process the commerce item attributes.

Create List Quote Item XA

protected ListOfQuoteItemXA createListQuoteItemXA (SiebelCommerceItem pCommerceItem) throws RepositoryException

This method processes all of the attributes on the commerce item and creates a quote item XA (i.e. a Siebel attribute) for each. The ATG to Siebel attribute mappings are as set out in Appendix C.

Convert Siebel Quote To Order

protected Order convertSiebelQuoteToOrder(Quote pQuote, String pProfileId, String pCatalogKey) throws TransformException

This method manages the conversion of a Siebel quote to an ATG order. It starts by creating the ATG order.

The method populateOrderPricingInformation is called to extract the quote level pricing information from the quote and to use this to populate the ATG order level pricing.

Finally the method convertQuoteItems is called to process all of the line items on the quote and convert them into ATG commerce line items.

Populate Order Pricing Information

public void populateOrderPricingInformation(Quote pQuote, Order pOrder)

This method populates the order level pricing information by using the information extracted from the Siebel quote. The field to property mappings are as set out in Appendix C.

Convert Quote Items

protected List<SiebelCommerceItem> convertQuoteItems (List<QuoteItem> pQuoteItems, String pCatalogKey) throws CommerceException

This method manages the conversion of the Siebel quote items to the corresponding ATG commerce items.

The basic conversion is achieved by calling the methods createCommerceItemFromQuoteItem and convertQuoteItemAttributes. The method convertQuoteItems is called to convert the quote item's children in ATG child commerce items. The child commerce items are then added to the parent.

The remaining processing is in deciding how to create the commerce item tree from the quote item hierarchy.

If the quote item is a promotion then the corresponding commerce item is added to a promotions list to be processed later.

If the quote item is a root product that is part of a bundle, then the corresponding commerce item is added to a promotions root products list to be processed later. If not, then the commerce item is added to the list to be returned.

When all quote items have been processed we check if a promotion was found during the processing. If it was then we need to add the root product commerce items to that promotion.

Finally the listed of converted commerce items is returned complete with the correct hierarchy.

Create Commerce Item From Quote Item

protected SiebelCommerceItem createCommerceItemFromQuoteItem (QuoteItem pQuoteItem, String pCatalogKey) throws CommerceException

This method creates an ATG commerce item from the specified Siebel quote item. It calls createItemPriceInfo to map the price information and then it calls the commerce item manager method, createCommerceItem, to create the actual commerce item. Finally it sets the action code on the newly created commerce item.

All field mappings are per those set out in Appendix C.

Convert Quote Item Attributes

protected void convertQuoteItemAttributes(QuoteItem pItem, SiebelCommerceItem pCommerceItem) throws CommerceException

This method converts Siebel quote item attributes into ATG attributes. The method createSiebelAttribute is called on the attribute manager to create the attribute and the method addAttributeToItem is called on the commerce item manager to add the attribute to the commerce item.

All field mappings are as set out in Appendix C.

Create Item Price Info

protected ItemPriceInfo createItemPriceInfo(QuoteItem pQuoteItem) throws CommerceException

This method creates an item price info from the quote item. This is set on the commerce item as the line item pricing. The field mappings are as set out in Appendix C.

Convert To Siebel Action Code

public String convertToSiebelActionCode (String pActionCode)

This method converts an ATG action code to the values required by the Siebel web services. The mappings are as set out in Appendix C.

Convert To ATG Action Code

public String convertToATGActionCode(String pActionCode)

This method converts from the Siebel action code to the ATG action code. Again the mappings are defined in Appendix C.

Is Promotion

protected boolean isPromotion(QuoteItem pQuoteItem)

This method checks if the product referenced in the Siebel quote item, is a promotion. By calling the getProductTypeCode method on the quote item we can check if it is set to ‘Promotion’. If it is then it’s a promotion.

Is Root Product

protected boolean isRootProduct (QuoteItem pQuoteItem)

This method establishes if the product referenced in the quote item is a root product. To do this we call the method getParentQuoteItemId on the quote item. If that is empty then it is a root product.

Is Part Of Bundle

protected boolean isPartOfBundle(QuoteItem pQuoteItem)

This method establishes if the product referenced in the quote item is part of a promotion bundle. It is part of a bundle if it is not a promotion and at least one of the method calls getProdPromName, getProdPromId or getProdPromInstanceId returns a non-empty value.

Create Siebel Attribute

public SiebelAttribute createSiebelAttribute(String pAttributeDefinitionId, String pAttributeName, String pActionCode, String pValue) throws CommerceException

This method creates a Siebel attribute in the repository.

OrderTools Component

The OrderTools component (/atg/commerce/order/OrderTools.properties) is redefined to point at the new extension class.

The various properties are changed to take account of the order repository extensions.

The layered component definition has the following entries:

```
$class=atg.siebel.order.SiebelOrderTools

commerceItemTypeClassMap+=\
   default=atg.siebel.order.SiebelCommerceItem,\
   siebelCommerceItem=atg.siebel.order.SiebelCommerceItem,\
   siebelAttribute=atg.siebel.order.SiebelAttribute

defaultCommerceItemType=siebelCommerceItem

beanNameToItemDescriptorMap+=\
   atg.siebel.order.SiebelCommerceItem=siebelCommerceItem,\
   atg.siebel.order.SiebelAttribute=siebelAttribute,\
   atg.siebel.pricing.SiebelItemPriceInfo=siebelItemPriceInfo

commerceItemManager=/atg/commerce/order/CommerceItemManager
siebelAttributeManager=/atg/commerce/order/SiebelAttributeManager
catalogTools=/atg/commerce/catalog/CatalogTools
priceListManager=/atg/commerce/pricing/priceLists/PriceListManager

siebelAttributeItemDescriptorName=siebelAttribute
userSiebelAccountIdPropertyName=siebelAccountId
organizationParentOrganizationPropertyName=parentOrganization
```

organizationNamePropertyName=name

organizationSiebelIdPropertyName=siebelId

idPropertyName=id

priceListPropertyName=priceList

defaultPriceListIdPropertyName=defaultPriceListId

addMRCToTotalFlag=false

Commerce Item Manager Extensions

This chapter details the required extensions to the commerce item manager.

SiebelCommerceItemManager Class

This section details the new SiebelCommerceItemManager class.

Class Declaration

The SiebelCommerceItemManager class (/atg/siebel/order) is an extension of the ootb CommerceItemManager class (/atg/commerce/order). The new class declaration is:

public class SiebelCommerceItemManager extends CommerceItemManager

Class Data

The class must maintain the following data:

- A property that references the SiebelAttributeManager component.

Class Implementation

The following methods are implemented in the class:

Create Commerce Item

This method has the following signature:

public CommerceItem createCommerceItem (String pItemType, String pCatalogRefId, Object pCatalogRef, String pProductId, Object pProductRef, long pQuantity, String pCatalogKey, String pCatalogId, String pSiteId, ItemPriceInfo pPriceInfo) throws CommerceException

This method over-rides the ootb commerce item manager's corresponding method. It is similar to the ootb method except that it reflects the fact that Siebel only uses products, and not SKUs.

Add Item To Order

public CommerceItem addItemToOrder (Order pOrder, CommerceItem pItem) throws CommerceException, InvalidParameterException

This method also over-rides the corresponding ootb method. It extensions ensure that line items are never merged but instead are always added as separate items. This reflects the fact that line items referencing the same product in Siebel may appear in more than one place in the order structure.

Add Attribute To Item

public void addAttributeToItem(SiebelCommerceItem pItem, SiebelAttribute pAttribute) throws CommerceException, InvalidParameterException

This method adds an attribute to the commerce item by invoking the addSiebelAttribute method on that commerce item.

Remove Item From Order

public void removeItemFromOrder(Order pOrder, String pCommerceItemId) throws CommerceException

This method over-rides the corresponding method from commerce item manager. It handles the removal of the commerce item tree.

CommerceItemManager Component

The CommerceItemManager component (/atg/commerce/order/CommerceItemManager.properties) is redefined to point at the new extension class. The new component definition has the following entries:

$class=atg.siebel.order.SiebelCommerceItemManager

siebelAttributeManager=/atg/siebel/order/SiebelAttributeManager

Siebel Attribute Manager

This chapter details the new Siebel attribute manager component and class. This manages commerce item attributes.

SiebelAttributeManager Class

This section details the new SiebelAttributeManager class.

Class Declaration

The SiebelAttributeManager class (/atg/siebel/order) is a new class. The class declaration is:

public class SiebelAttributeManager extends GenericService

Class Data

The class must maintain the following data:

- A property that references the OrderTools component.

Class Implementation

The following methods are implemented in the class:

Create Siebel Attribute

This method has the following signature:

public SiebelAttribute createSiebelAttribute () throws CommerceException

This method manages the creation of a default Siebel attribute by calling the createSiebelAttribute method with null values for each parameter.

Create Siebel Attribute

This method has the following signature:

public SiebelAttribute createSiebelAttribute (String pAttributeDefinitionId, String pActionCode, String pValue) throws CommerceException

This method manages the creation of a Siebel attribute using the specified parameters to set the property values. The actual attribute is created with a call to the order tools method createSiebelAttribute.

SiebelAttributeManager Component

The SiebelAttributeManager component (/atg/siebel/order/SiebelAttributeManager.properties) is defined as follows:

$class=atg.siebel.order.SiebelAttributeManager

orderTools=/atg/commerce/order/OrderTools

Item Pricing Engine

This chapter details the over-rides required to item pricing engine to ensure that item pricing is done by Siebel instead ATG.

SiebelItemPricingEngine Class

This section details the new SiebelItemPricingEngine over-ride class.

Class Declaration

The SiebelItemPricingEngine class (/atg/siebel/pricing) is a new class that over-rides the existing ItemPricingEngine class. The class declaration is:

public class SiebelItemPricingEngine extends PricingEngineService implements ItemPricingEngine

Class Data

The class must maintain the following data:

- A property that references the OrderTools component.
- A property that references the WebServiceHelper component.

ItemPricingEngine Implementation

The following methods must be implemented from the ItemPricingEngine interface:

Price Items

public List priceItems (List pItems, Collection pPricingModels, Locale pLocale, RepositoryItem pProfile, Order pOrder, Map pExtraParameters) throws PricingException

This method calls the order tools method convertOrderToSiebelQuote to get the quote format from the ATG order. The quote format is required to be passed to the Siebel pricing services.

Then the method calls siebelDynamicPricing, which manages the interaction with the Siebel web service to do item pricing. That method returns a list of SiebelItemPriceInfo objects that are returned to the caller.

Price Item

public ItemPriceInfo priceItem (CommerceItem pItem, Collection pPricingModels, Locale pLocale, RepositoryItem pProfile, Map pExtraParameters) throws PricingException

This is a null implementation.

Price Each Item

public List priceEachItem (List pItems, Collection pPricingModels, Locale pLocale, RepositoryItem pProfile, Map pExtraParameters) throws PricingException

This is a null implementation.

Class General Implementation

The following methods provide the new pricing functionality:

Siebel Dynamic Pricing

public List<SiebelItemPriceInfo> siebelDynamicPricing (Quote pQuote) throws PricingException

This method calls the Siebel CalculatePrice web service (via the callWebService method), passing the quote as a parameter.

The web service returns a list of quote items that are then processed. Each quote item is converted into a Siebel item price info object and the resulting list is passed to the caller.

Call Web Service

protected CalculatePriceOutput callWebService (CalculatePriceInput pInput)

This is the method that binds to the web service and does the actual call.

ItemPricingEngine Component

The ItemPricingEngine component (/atg/commerce/pricing/ItemPricingEngine.properties) is redefined to point at the new over-ride class.

$class=atg.siebel.pricing.SiebelItemPricingEngine

orderTools=/atg/commerce/order/OrderTools

webServiceHelper=/atg/siebel/integration/WebServiceHelper

priceInfoClass=atg.siebel.pricing.SiebelItemPriceInfo

Appendix A: Siebel Quote Format

This appendix gives more detail on the Siebel Quote logical entities.

Quote

The Quote entity is the container for the Siebel quote.

Field	Description
QuoteNumber	The number of the quote.
AccountId	Id of the shopper's parent account.
Account	Name of the shopper's parent account.
PriceListId	The price list to be used.
Revision	Quote revision. Set to "1".
MRCTotal	Monthly recurring charge total.
NRCTotal	Non recurring charge total.
ListOfQuoteItem	List that contains the quote line items. Quote line items are held in the QuoteItem element (see below)

Quote Item

The QuoteItem contains a quote line item.

Field	Description
Id	Id of the quote item. Set to the commerce item id to make it unique.
IntegrationId	Integration Id of the quote Item. Corresponds to ATG commerce item id.
ProductId	Id of the product being purchased.
Quantity	Quanitity of the product being purchased.
ActionCode	Specifies the action to be taken on the quote Item. The values are: - <blank> is 'Add' item. - Minus sign (i.e. '-') is 'NoAction'. - Update is an 'Update' action. - Delete is a 'Delete' action.
ListPrice	The list price for the product.
CurrentPrice	The currently calculated price for the product, i.e. after it has been priced by Siebel.
DiscountAmount	The amount of discount applied to the product during pricing.

MRCCxTotal	Monthly recurring charges relating to the line item.
NRCCxTotal	Non-recurring charges relating to the line item.
CurrencyCode	Code representing the currency used in pricing.
PrePickCD	Set to “Y”.
AccountId	Id of the shopper’s parent account.
ListOfQuoteItem	List that contains child line items for this line item. This represents a parent child relationship between QuoteItems.
ListOfQuoteItemXA	List of line item attributes. Each attribute is held in a QuoteItemXA element.

Quote Item XA

The QuoteItemXA contains a Siebel attribute.

Field	**Description**
Attribute	The attribute name.
ActionCode	Specifies the action to be taken on the QuoteItemXA element (i.e. the attribute). The values are: - <blank> is Add attribute. - Minus sign (i.e. ‘-‘) is NoAction. - Update is an update action. - Delete is a delete action.
Value	Attribute’s value.

Appendix B: ATG Order Format

The first section of this appendix (ATG Order) describes the logical entities and properties (fields) which exist in the out-of-the-box ATG order repository version 10.2. Only the entities and properties that are relevant to the ATG – Siebel order integration are included in the descriptions below.

The second section (ATG Order Extensions For Siebel) describes the order repository extensions required to support order conversion to the Siebel Quote format.

ATG Order

The tables below describe the ATG order repository (for ATG 10.2). See reference 3 for more details on ATG orders.

Order

The order item descriptor from the ATG order repository represents an order that includes the order items and the pricing, billing and shipping information.

PROPERTY (FIELD)	DESCRIPTION
id	Id of the order.
type	Order type. Defaults to 'order'.
version	Used to ensure that the order is no updated concurrently by two difference sources.
profileid	Id of the shopper's profile.
description	Description of the order.
state	Order state. Set to INCOMPLETE & SUBMITTED.
submittedDate	Date of submission to the OMS. In this case to Siebel.
lastModifiedDate	The date that the order was last modified.
priceInfo	An orderPriceInfo item that stores order level pricing information. It is a sub type of amountInfo.
taxPriceInfo	A taxPriceInfo item that stores order tax information. It is a sub type of amountInfo.
shippingGroups	One or more shippingGroup items that store shipping details.
paymentGroups	One or more paymentGroup items that store billing information.
commerceItems	One or more commerceItems that store the items in the shopping cart / order.

Amount Info

The amountInfo item descriptor is the parent for all of the various pricing details, such as order pricing, item pricing, tax pricing etc.

PROPERTY (FIELD)	DESCRIPTION
type	The type of the amountInfo item. It can be related to the order, an item, tax or shipping.
currencyCode	The currency associated with the pricing information.
amount	The amount.
discounted	Flag to indicate whether or not the order contains discounts.
amountIsFinal	Flag to indicate that the amount is frozen and shall not be changed.
adjustments	A collection of pricingAdjustment items that list the adjustments made to the order pricing.

Order Price Info

The orderPriceInfo item descriptor is a sub type of amountInfo. It is used to store order level pricing information.

PROPERTY (FIELD)	DESCRIPTION
rawSubTotal	The sub total without tax and shipping.
Tax	The total tax on the order.
Shipping	The shipping costs for the order.

Tax Price Info

The taxPriceInfo item descriptor is also a sub type of amountInfo. It is used to store order level tax information.

PROPERTY (FIELD)	DESCRIPTION
cityTax	Tax levied at the city level.
countyTax	Tax levied at the county level.
stateTax	Tax levied at the state level.
countryTax	Tax levied at the country level.

Item Price Info

The itemPriceInfo item descriptor is also a sub type of amountInfo. It is used to store order line item pricing information.

PROPERTY (FIELD)	DESCRIPTION
listPrice	List price for the product.

rawTotalPrice	Total cost of the full quantity of product.
salePrice	Sale price for the product.
onSale	Flag to indicate whether or not the product is on sale.
orderDiscountShare	The amount of the total order discount attributed to this commerce item.
quantityDiscounted	The number of items discounted within this commerce item.
pricelist	The price list reference.
currentPriceDetails	A collection of objects that store the detailed pricing information for all of the items in this commerce item.

Shipping Group

The shippingGroup item stores shipping details for the order. It also has sub types for hard goods and electronic goods.

PROPERTY (FIELD)	DESCRIPTION
Type	The shipping group type. Usually set to hardgoodShippingGroup for hard goods and to electronicShippingGroup for electronic goods.
shippingMethod	Method by which the goods are to be shippied.
Description	A description of the shipping group.
State	The state of the shipping group.
submittedDate	The date that the shipping group was submitted to the OMS.
priceInfo	Costs associated with the shipping group.
Order	Reference to the parent order.
handlingInstructions	A collecton of instructions for handling the goods.
specialInstructions	A collection of other instructions for shipping.

Hard Good Shipping Group

The hardgoodShippingGroup extends the shippingGroup item to add a physical address and other properties.

PROPERTY (FIELD)	DESCRIPTION
trackingNumber	Tracking number from the shipping company.
Prefix	Name prefix.
firstName	Addressee's first name.
middleName	Middle name.
lastName	Last name.
Suffix	Suffix.
jobTitle	Job title.
companyName	Name of the company.

address1	First line of the address.
address2	Second line of the address.
address3	Third line of the address.
city	City.
county	County.
postalCode	Post code or Zip code.
country	Country.
phoneNumber	Phone number.
faxNumber	Fax number.
email	Email address.

Payment Group

The paymentGroup item descriptor stores billing information for the order.

PROPERTY (FIELD)	DESCRIPTION
type	The type of the payment group. The type 'creditCard' is the only type supported for this integration.
paymentMethod	The method of payment used.
amount	The amount to be paid.
amountAuthorized	Amount already authorized.
amountDebited	Amount debited.
amountCredited	Amount credited.
currencyCode	The currency of the payment.
state	The payment group state.
submittedDate	The date the payment was submitted.
order	Order reference.
authorizationStatus	Authorization status.
debitStatus	Debit status.
creditStatus	Credit status.

Credit Card

The creditCard item extends the base paymentGroup to add credit card details and billing address details.

PROPERTY (FIELD)	DESCRIPTION
creditCardNumber	Credit card number.
creditCardType	The type of the credit card, e.g. Visa, Mastercard etc.

expirationMonth	Expiration month.
expirationDayOfMonth	Expiration day of the month.
expirationYear	Expiration year.
prefix	Name prefix.
firstName	Payer's first name.
middleName	Middle name.
lastName	Last name.
suffix	Suffix.
jobTitle	Job title.
companyName	Name of the company.
address1	First line of the address.
address2	Second line of the address.
address3	Third line of the address.
city	City.
county	County.
postalCode	Post code or Zip code.
country	Country.
phoneNumber	Phone number.
faxNumber	Fax number.
email	Email address.

Commerce Item

The order price info item stores pricing information at the order level.

PROPERTY (FIELD)	DESCRIPTION
Type	Commerce item type.
catalogId	Id of the catalog to be used.
catalogRefId	Refers to the item in the catalog, e.g. productId or skuId.
catalogKey	Key identifying the catalog to use. The key is determined by the user's locale and the value is the corresponding repository to use (for example, en_US=ProductCatalog, fr_FR=FrenchCatalog).
productId	Id of the product.
siteId	The site id of the site where the product was added to the order.
quantity	The quantity of the product piurchased.
state	The state of the commerce item.
priceInfo	An itemPriceInfo object containing the pricing information for the product.
order	Reference to the parent order.

ATG Order Extensions for Siebel

This section details the ATG order extensions required to support conversion to the Siebel Quote format.

Commerce Item

This item descriptor corresponds to the Siebel logical entity named QuoteItem. The Commerce Item item descriptor already exists in ATG.

The integration doesn't require any extensions to the physical schema for this item descriptor but we do need to add a new sub type option to the type property. The new option is named 'siebelCommerceItem'.

We also need to change the item-type of the priceInfo property to be "siebelItemPriceInfo" (see below for more details). This is because we are extending the itemPriceInfo item descriptor to hold some additional Siebel pricing information, i.e. non-recurring and monthly recurring pricing.

Siebel Commerce Item

This item descriptor also corresponds to the Siebel logical entity named QuoteItem. This is a new logical entity in ATG and it is a sub type of the Commerce Item entity above.

The following table describes the item descriptor attributes.

Item Descriptor Attribute	Value
Name	siebelCommerceItem
Table Name	SBL_ORD_COMM_ITEM
Table Type	auxiliary
Item Cache Size	1000
Query Cache Size	1000
Id Space Name	

The following table describes the item descriptor properties.

Property Name	Data Type	Component Item Type	Column Name	Table Name / Id Column / Type (If different from main table)
Id	string		COMMERCE_ITEM_ID	
actionCode	enumerated		ACTION_CODE	
attributes	list	siebelAttribute	ATTRIBUTE_ID	SBL_ORD_COMM_ITEM_ATTR COMMERCE_ITEM_ID multi
commerceItems	list	siebelCommerceItem	COMMERCE_ITEM_CHILD_ID	SBL_ORD_COMM_ITEM_CHILD

				COMMERCE_ITEM_ID multi

The enumerations for the actionCode property are:

<option value="NoAction" code="0"/>

<option value="Add" code="1"/>

<option value="Update" code="2"/>

<option value="Delete" code="3"/>

Amount Info & Item Price Info

The Item Price Info item descriptor corresponds to the Siebel logical entity named QuoteItem. The Item Price Info extends the Amount Info item descriptor and both already exist in ATG. The Item Price Info item descriptor needs to be extended to hold the additional Siebel pricing information.

To extend the Item Price Info item descriptor we must add a new option to its parent item descriptor, Amount Info. We then create the new item descriptor Siebel Item Price Info (see below) and point it's super-type attribute at the Item Price Info item descriptor.

Siebel Item Price Info

This item descriptor also corresponds to the Siebel logical entity named QuoteItem, since it extends the Item Price Info item descriptor (see above).

The values here are returned in a QuoteItem as a result of a Calculate Price web service call to Siebel. This is a new logical entity in ATG and it is a sub type of the Item Price Info entity above.

The following table describes the item descriptor attributes.

Item Descriptor Attribute	**Value**
Name	siebelItemPriceInfo
Table Name	SBL_ITEM_PRICE_INFO
Table Type	auxiliary
Item Cache Size	1000
Query Cache Size	1000
Id Space Name	

The following table describes the item descriptor properties.

Property Name	**Data Type**	**Component Item Type**	**Column Name**	**Table Name / Id Column / Type** **(If different from main table)**
nonRecurringPrice	double		NON_RECURRING_PRICE	

monthlyRecurringPrice	double		MONTHLY_RECURRING_PRICE	

Siebel Attribute

This item descriptor also corresponds to the Siebel logical entity named QuoteItemXA. This is a new logical entity in ATG.

The following table describes the item descriptor attributes.

Item Descriptor Attribute	Value
Name	siebelAttribute
Table Name	SBL_ORD_ATTR
Table Type	primary
Item Cache Size	1000
Query Cache Size	1000
Id Space Name	siebel-order-commerce-item-attribute

The following table describes the item descriptor properties.

Property Name	Data Type	Component Item Type	Column Name	Table Name / Id Column / Type (If different from main table)
Id	string		ATTRIBUTE_ID	
attributeDefinitionId	string		ATTRIBUTE_DEF_ID	
attributeName	string		ATTRIBUTE_NAME	
actionCode	enumerated		ACTION_CODE	
value	string		VALUE	

Appendix C: Siebel – ATG Order Mappings

This chapter details the field level mappings between the ATG and Siebel logical entities.

Base Mappings

This section details the base mappings. Each sub section heading represents a Siebel logical entity as defined in the format set out in Appendix A.

Each shaded row indicates a complex mapping. Complex mappings are those that do not have a simple 1:1 mapping between the Siebel field and the ATG property. Each complex mapping has a corresponding section in the Complex Mappings section.

Quote

Siebel Field Name	ATG Item Descriptor	ATG Property Name
QuoteNumber	order	id
AccountId	order	profileId.siebelAccountId
Account	order	profileId.parentOrganization.name
PriceListId	order	profileId.priceList
MRCTotal	order	priceInfo.amount
NRCTotal	order	priceInfo.amount
ListOfQuoteItem	order	commerceItems

Quote Item

Siebel Field Name	ATG Item Descriptor	ATG Property Name
Id	siebelCommerceItem	id
IntegrationId	siebelCommerceItem	id
ProductId	siebelCommerceItem	catalogRefId, productId
Quantity	siebelCommerceItem	quantity
ActionCode	siebelCommerceItem	actionCode
ListPrice	siebelCommerceItem	priceInfo.listPrice
CurrentPrice	siebelCommerceItem	priceInfo.amount
DiscountAmount	siebelCommerceItem	priceInfo.orderDiscountShare
MRCCxTotal	siebelCommerceItem , siebelItemPriceInfo	priceInfo.monthlyRecurringCharges

NRCCxTotal	siebelCommerceItem , siebelItemPriceInfo	priceInfo.nonRecurringCharges
CurrencyCode	siebelCommerceItem	priceInfo.currencyCode
AccountId	siebelCommerceItem	profileId.siebelAccountId
ListOfQuoteItem	siebelCommerceItem	commerceItems
ListOfQuoteItemXA	siebelCommerceItem	Attributes

NOTE: Siebel ActionCodes must be converted to ATG actionCodes. If the Siebel code is blank then the ATG code is 'Add'; if the Siebel code is the minus sign then the ATG code is 'NoAction'; if the Siebel code is either 'Update' or 'Delete' then ATG uses the same value.

Quote Item XA

Siebel Field Name	ATG Item Descriptor	ATG Property Name
Attribute	siebelAttribute	attributeDefinitionId
ActionCode	siebelAttribute	actionCode
Value	siebelAttribute	value

NOTE: ActionCode conversion is the same as for Quote Items above.

Complex Mappings

The sections below specify the Siebel to ATG order mappings that are not simple 1:1 mappings between Siebel fields and ATG properties. Each sub section heading relates to a Siebel entity.

Quote

AccountId, Account & PriceListId

Each of these fields maps onto an ATG profile. The ATG order references a profile but only by profileId. That means that the profileId must be used to query the profile repository in order to get the corresponding profile item. Use the profile item to set the Siebel fields as per the mapping table above.

MRCTotal & NRCTotal

A new orderPriceInfo item must be created. The new item is referenced by the priceInfo property. By default the orderPriceInfo's amount property is set to NRCTotal, but this must be configurable so that it could be set to NRCTotal + MRCTotal.

ListOfQuoteItem

This element in Siebel contains a list of QuoteItem elements. A siebelCommerceItem item must be created for each QuoteItem element and a reference to each of these is stored in the commerceItems collection property.

Quote Item

ListPrice, CurrentPrice, DiscountAmount & CurrencyCode

A new itemPriceInfo item must be created. The new item is referenced by the priceInfo property. Set the itemPriceInfo properties as specified by the mapping table.

AccountId

This field maps onto an ATG profile. The ATG order references a profile only by profileId. That means that the profileId must be used to query the profile repository in order to get the corresponding profile item. Use the profile item to set the Siebel field as per the mapping table above.

ListOfQuoteItem

This element in Siebel contains a list of QuoteItem elements. In this case these are children of the current QuoteItem. A siebelCommerceItem item must be created for each QuoteItem element and a reference to each of these is stored in the commerceItems collection property.

ListOfQuoteItemXA

This element in Siebel contains a list of QuoteItemXA elements. A siebelAttribute item must be created for each QuoteItemXA element and a reference to each of these is stored in the attributes collection property.