Solve Inter-Company Transfer Pricing Challenges Using Oracle Advanced Pricing

Peter Belter

Abstract

This presentation will discuss setups needed in Oracle Advanced Pricing to price intercompany (IC) orders based on the cost of the item and variable markups, dependent on the origin, destination of the transfer and product type.

Introduction

Oracle Advanced Pricing provides a powerful, flexible and extendable engine to derive prices based on business conditions specific to industry and business model of the customer. While many companies take advantage of this tool to tailor their customer prices, few use it to maintain their intercompany margins and manage tax liabilities. It is a little known feature of Advanced Pricing which, with some setups and minimal or no customizations, can be used to maintain cash flow and manage tax liabilities between different businesses under the same corporate umbrella.
Overview of common practices

Several companies create static intercompany price lists by using the ‘Set List Price Equal to Cost Form’, then adjusting it by specific markup using the ‘Adjust Price List’ functionality.

A price list created this way is then attached to:

- the Shipping networks.
- the internal customer site for intercompany invoicing driven by internal drop shipments.

While it works, this approach has several significant disadvantages:

- Can be only used for Standard Costing inventories but not for Average, LIFO, FIFO or Actual costing
- Prices have to be manually adjusted when costs change. New price lists may be built for new costs but that creates unnecessary maintenance
- As new items are added they have to be manually added to the price list
- Static price list have to account for all possible item transfers between all organizations creating very large volume of data and significant overheads as shown below.

<table>
<thead>
<tr>
<th>Number of subsidiaries</th>
<th>Number of items</th>
<th>Number of price lists</th>
<th>Total price list lines</th>
<th>With extra IC node</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10,000</td>
<td>2</td>
<td>20,000</td>
<td>40,000</td>
</tr>
<tr>
<td>4</td>
<td>20,000</td>
<td>12</td>
<td>240,000</td>
<td>2,880,000</td>
</tr>
<tr>
<td>6</td>
<td>20,000</td>
<td>30</td>
<td>600,000</td>
<td>18,000,000</td>
</tr>
<tr>
<td>8</td>
<td>50,000</td>
<td>56</td>
<td>2,800,000</td>
<td>156,800,000</td>
</tr>
<tr>
<td>10</td>
<td>100,000</td>
<td>90</td>
<td>9,000,000</td>
<td>810,000,000</td>
</tr>
<tr>
<td>50</td>
<td>250,000</td>
<td>2450</td>
<td>612,500,000</td>
<td>1,500,625,000,000</td>
</tr>
</tbody>
</table>
To avoid manual maintenance some businesses create customizations to maintain prices in the intercompany price list(s). In almost every case those customizations turn out to be very labor and maintenance intensive, especially in exception handling. The number of intercompany price list lines can easily surpass the number of customer lines in customer price lists, slowing down their performance.
Using Oracle Advanced Pricing to drive intercompany pricing

Overview of needed setups

<table>
<thead>
<tr>
<th>Setup</th>
<th>Required/Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shipping Networks</td>
<td>required</td>
</tr>
<tr>
<td>Intercompany Transaction Flows</td>
<td>required</td>
</tr>
<tr>
<td>Create custom responsibility for Intercompany pricing</td>
<td>optional but highly recommended for easiness of use</td>
</tr>
<tr>
<td>Profile options</td>
<td>required</td>
</tr>
<tr>
<td>Mapping cost as price, converted to transaction currency</td>
<td>required</td>
</tr>
<tr>
<td>Create dynamic formula to calculate cost in price list currency</td>
<td>required</td>
</tr>
<tr>
<td>Map ALL Items for IC Pricing</td>
<td>required</td>
</tr>
<tr>
<td>Create Intercompany Price list</td>
<td>required</td>
</tr>
<tr>
<td>Map additional pricing attributes</td>
<td>optional</td>
</tr>
<tr>
<td>Update INTCOM pricing entity</td>
<td>optional, highly recommended</td>
</tr>
<tr>
<td>Create Markup Modifiers</td>
<td>optional, but usually needed</td>
</tr>
<tr>
<td>Map additional qualifiers</td>
<td>optional, required if logical intercompany transaction nodes are present</td>
</tr>
<tr>
<td>Custom pricing phases</td>
<td>optional</td>
</tr>
</tbody>
</table>
Shipping networks and Intercompany Transaction Flows

Setting up the shipping Networks and Intercompany transaction flows is outside of scope of this document. One feature worth mentioning is the ability to select the currency the resulting intercompany invoice.

Regardless of the value selected above. The price of the intercompany invoice will be always calculated using the USD intercompany price list, and then converted to the relevant currency of the intercompany invoice as a post-pricing event. This allows maintaining only one intercompany price list in USD, even for transfers between 2 entities where neither uses USD.
Create Custom Responsibility

To begin with, we recommend creating separate responsibility for Intercompany pricing, to keep the IC and Sales price lists, modifiers, formulas, qualifiers and modifiers separate between the Order Management and Intercompany Pricing Entities. This is not required, as alternatively you can set up the relevant profile options user level for a specially created user, separate responsibility is by far a cleaner solution.

Go to Application Developer -> Profiles and check the responsibility checkboxes for the below:
If you are on a new installation of R12 done after 2009, those values are already checked.

Now create your custom responsibility

(N) System Administrator -> Security -> Responsibility -> Define

Add the responsibility to your user.

You have successfully prepared your workshop for working with Intercompany Pricing.
**Profile Options**


![System Profile Values](image)

Set the following profiles on the site level:

- CST: Transfer Pricing Option - Yes, Price As Incoming Cost (read more about other settings)
- INV: Advanced Pricing for Inter-Org Transfers - Yes
- INV: Advanced Pricing for Intercompany Invoice - Yes
- INV: Always suffix inter-Company AP Invoice number – Yes (optional)
- INV: Inter-Organization Currency Conversion – Corporate
- INV: Intercompany Currency Conversion - Corporate
- INV: Intercompany Invoice for Internal Orders – Yes (optional)
Mapping cost as price, converted to transaction currency

Log into the Intercompany Pricing responsibility -> Setup -> Attribute Management -> Attribute Linking and Mapping. Select Intercompany Transaction as the Pricing Entity and Pricing Context, then map new COST attributes for item cost and for converting the cost to the transaction currency.

---

[Diagram showing the setup process for mapping cost attributes]

---

[Table showing relevant context and attributes]

---

[Diagram showing the context setup for COST (Cost Context)]
In the ‘Global Object Name’ field you can see the name of the record structure that has all the variables that used when deriving the price. In this case it is INV_IC_ORDER_PUB.G_LINE.
Both INTORG and IC request types need to be mapped, to manage the material transaction flow
(INTORG) and the IC AR and AR invoicing (IC)

**Note:** Check the INV_IC_ORDER_PUB package definition to see available fields for the G_LINE record.

To get the item cost use the standard Oracle API cst_cost_api.get_item_cost. The actual expression to
be pasted into the User Value String of the ITEM_COST is:

```sql
cst_cost_api.get_item_cost(1, INV_IC_ORDER_PUB.G_LINE.INVENTORY_ITEM_ID,
INV_IC_ORDER_PUB.G_LINE.SHIP_FROM_ORG_ID,NULL,NULL)
```

The cost is usually not enough, unless all of your operations are done in single currency. In case you have
warehouses in countries with other currencies the cost will be expressed in the currency of the
operating unit the warehouse belongs to. Assuming that all of our intercompany prices are always
calculated in USD as per our recommendation, we only need to find the currency of the shipping
organization. It can be found the standard API call:

```sql
csd_cost_analysis_util.get_glcurrencycode(p_operating_unit_id).
```

At this point we do not have the operating_unit_id. It cannot be simply taken from the order or as
INV_IC_ORDER_PUB.G_LINE.ORG_ID because warehouse can belong to different OU than the OU of the
transaction. In example in the Internal Drop Shipment flow the warehouse listed on the order belonging
to the Selling OU, belongs to the Shipping OU. The OU of the warehouse can be derived from the
SHIP_TO_ORG_ID using the standard API:

```sql
qa_moac_pkg.derive_ou_id(p_ship_from_org_id)
```

After combining those 2 APIs the expression is:

```sql
  csd_cost_analysis_util.get_glcurrencycode(
      qa_moac_pkg.derive_ou_id (INV_IC_ORDER_PUB.G_LINE.SHIP_FROM_ORG_ID)
  )
```

Now that the currency of the shipping warehouse is known it can be converted to the currency of the
transaction with the conversion API:

```sql
gl_currency_api.get_rate(p_from_currency, p_to_currency, p_conversion_date, p_conversion_type)
```

Combining all 3 APIS we get the full expression to be pasted to the User Value String field of
CONV_RATE:

```sql
  gl_currency_api.get_rate (p_from_currency, p_to_currency, p_conversion_date, p_conversion_type)
```

Copyright ©2015 by Peter Belter
Assuming we are using 'USD' for all intercompany transactions and convert other currency values with 'Corporate' rate

This will work on R12 as long as you don’t use Global Purchase Agreements and none of the warehouses involved in the transaction is an OPM organization. In those cases you will need to use additional logic, which I am omitting here to keep the case simple. For setup in OPM organizations review Appendix 1. For intercompany transactions triggered by Global Purchasing Agreements review Appendix 2

Additional coding is needed in 11i where the qa_moac_pkg.derive_ou_id function does not exist.

When mapping is completed, run the Build Attribute Mapping Rules Concurrent request.
Create dynamic formula

With cost and conversion rate mapped, a dynamic pricing formula can be created to calculate the cost standardized to single currency (USD).

The formula multiples the cost pricing attribute times the currency conversion rate from the currency of the cost to the currency of the intercompany invoice, returning item cost in USD.

In this example I have added a 15% markup to the cost to be used as the intercompany price. A more flexible approach is to use the intercompany pricing modifiers to apply different markups based on qualifying conditions such as product line or the ship from and ship to organizations.

Remember to build the formula package after definition is done

Because this formula is created in the Intercompany Pricing responsibility, it is visible, but not updateable in the regular Oracle Pricing Manager or any OM responsibility. With the formula ready we can create the Intercompany Price List.
Map ALL Items for IC Pricing

For standard Order Fulfillment entity, Advanced Pricing allows price list line with formula for ALL ITEMS.

We will need to do the same for Intercompany Pricing entity but the Product Context for ALL ITEMS is not mapped out of the box. Here are the mapping setups:

Add the ALL_ITEMS context and click Attribute Mapping to map it

In addition to mapping the ALL_ITEMS attribute we recommend mapping individual segments of the item category. This is covered in Appendix 3
Paste the constant Value of ALL into the User Value String

It needs to be mapped for both the IC and INTORG request type.

With the ALL ITEMS product context mapped we can create our cost based intercompany price list.
Create Intercompany Price list

Using our customer Intercompany Pricing Responsibility create the ‘Intercompany Price List’

It needs to have as many lines as Primary units of measure in each unit of measure class used for intercompany transfers. Each line has the Product Attribute equal to ‘All Items’ and Product Value ‘ALL’ with price of each item calculated using our cost based dynamic pricing formula.

Note the Pricing Transaction Entity for your price list. Because it was created in our custom responsibility it will not be usable in Order Management or any other sales applications.

For intercompany pricing the List Line Base Price pricing phase used to find list price for an item has ‘Seeded Search Flag’ set to Yes.
This means that the intercompany price list specified on the Shipping Networks and on the Internal Customer site is ignored, but instead the qualifying price list with highest precedence is found. This also means that setting default price list on internal customer or shipping network is not needed... at least in theory. In practice you still have to enter a price list on the internal customer ship-to site to work around an unrelated bug. This is true for both 11i and R12. The price list entered on the internal customer site is ignored for pricing purposes, but the ‘Create Intercompany AR Invoices’ process will fail if the price list field is null.

You can change the flag to ‘No’ and then attach the intercompany price list to all Shipping Networks nodes and internal customer sites or, if this is your only intercompany price list, leave it as is. Check the value of the QP: Blind Discount Option. If it is set to ‘No’ the price list may need to have at least one qualifier that always evaluates to true in order to be considered...
Update INTCOM pricing entity

To prevent customer price lists from being used in Intercompany transactions update the INTCOM pricing entity by disabling the QP source systems.
Create Markup Modifiers

Selling items at cost is not common business practice and there is usually a surcharge markup associated with each transaction. If the markup is uniform across the board then we can incorporate it directly into the cost formula. If it varies depending on product type, source or destination organization, intercompany modifiers should be used.

The Intercompany modifiers are defined in our custom responsibility and therefore can be only used for intercompany transactions. There are not visible to customer orders.

Here is an example of an intercompany modifier

![Advanced Pricing - Define Modifier](image)

Created using our custom intercompany pricing responsibility, it is applicable only to Intercompany transactions.

![Advanced Pricing - Define Modifier](image)

The modifier will apply 5% surcharge on all laptops, 4% on all other computers and 2% for all other items. But only if
items are transferred from US to France (group 1) or French operating unit is selling items shipped from a warehouse belonging to the US operating unit (group 2).

Note that we have changed precedence on the qualifiers to 990 so that they do not interfere with the precedence on the lines. This can be done either for each modifier or updated on the qualifier sourcing definition to be always 990.

The markup for each set of counties or product groups can be easily adjusted on the modifier line, instead of adjusting thousands of individual prices.

If you have BEST PRICE incompatibility resolution set for your standard customer modifiers instead of PRECEDENCE, you need to create separate pricing phase for Intercompany modifiers. Refer to appendix 4 for details.

Other than using the optional mapping for Product Family and Product Class, this is all achieved with setups and no customization. The entire solution can be set up and tested in matter of days.
**Adjusting intercompany profits between nodes in multi node transfers**

Oracle Intercompany Transaction flows allow setting up multi-node transfers where goods physically shipped between 2 countries can change legal ownership across a number of intermediaries, each generating a set on intercompany AP and AR invoices.

Here is an example where sales from Germany to US always goes through France.

We can use this setup in conjunction with Advanced Pricing to shift profits to operating units in favorable tax jurisdictions.

In our example Vision Operations sells items to Vision France at cost while the average margin in France for customer sales is 30%.

Vision has European headquarters in Ireland partially because it has more favorable tax laws than France.

In our case study France has 30% income tax while Ireland has only 10%.
Today the sale is:

<table>
<thead>
<tr>
<th>US Cost</th>
<th>Sells to Ireland</th>
<th>Profit in IR</th>
<th>Tax in IR</th>
<th>Sells to France</th>
<th>France sells to Customer</th>
<th>Profit in FR</th>
<th>Tax in FR</th>
<th>Vision Gross Profit</th>
<th>Net profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$100</td>
<td>$100</td>
<td>$0</td>
<td>$0</td>
<td>$100</td>
<td>$130</td>
<td>$30.0</td>
<td>$9.0</td>
<td>$30</td>
<td>$21.0</td>
</tr>
</tbody>
</table>

With profit in Ireland adjusted to 25%:

<table>
<thead>
<tr>
<th>US Cost</th>
<th>Sells to Ireland</th>
<th>Profit in IR</th>
<th>Tax in IR</th>
<th>Sells to France</th>
<th>France sells to Customer</th>
<th>Profit in FR</th>
<th>Tax in FR</th>
<th>Vision Gross Profit</th>
<th>Net profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$100</td>
<td>$100</td>
<td>$25</td>
<td>$2.5</td>
<td>$125</td>
<td>$130</td>
<td>$5.0</td>
<td>$1.5</td>
<td>$30</td>
<td>$23.5</td>
</tr>
</tbody>
</table>

The tax liability was reduced by more than half, from $9 to $4 ($2.5+$1.5) and net profit increased by over 10%.

At the same time we have a factory in Ireland that produces variety of items at an average margin of only 20%. While in Ireland we want to markup the US made items at 25% we only want to markup the Irish made goods at 15%.

Here is how to set it up on the pricing side. This does not include the multi node item flow setups that need to be done separately:

- Define a new ‘Origin point’ qualifier. This is needed because at the time of the logical transfer from Ireland OU to France OU, Ireland is the Shipping OU and France is Selling OU while the information about the goods being sources from US is lost.
- We do not need modifier for transfers from US to Ireland. Since there is not markup the Intercompany price list is sufficient, but we need to define modifier for transfers from Ireland to France where the goods originated from US.
We also need markup for direct sales from Ireland to France.
Appendix 1

For organizations using Oracle Process Manufacturing, mapping of the item cost attribute needs to be done using OPM API since the standard discrete cost API does not return cost for process organizations. If you are using both OPM and discrete organizations you will need to NVL both costs in the User Value Sting expression so that the appropriate cost is derived.

The Cost API for OPM is

```
GMF_CMCOMMON.get_cmpt_cost( INV_IC_ORDER_PUB.G_LINE.INVENTORY_ITEM_ID,
INV_IC_ORDER_PUB.G_LINE.SHIP_FROM_ORG_ID, sysdate, P_cost_type_id ,0)
```

Where `P_cost_type_id` needs to be substituted by value received from:

```
select cost_type_id
from CM_MTHD_MST
where cost_mthd_desc='Standard Cost'
```

This value may be different for each implementation so we are not providing it directly in the expression for `GMF_CMCOMMON.get_cmpt_cost` but it needs to be typed in based on the value from query above.

Sysdate is the date of the cost. Many OPM organizations do not calculate their cost until next month after transactions. This can lead to situations where the intercompany invoicing done using the above API call will result in all prices being 0, which can be only undone with a manual entry in GL and is a very messy and laborious process to fix. To prevent that one may consider substituting sysdate with a date that already has costs calculated, in example the date of the last closed accounting period.

In enterprises where both OPM and discrete organizations exist, we recommend mapping the OPM and Discrete costs into separate pricing attributes then use an NVL statement in the pricing formula or qualified price list to select the correct cost. This approach gives you more transparency when troubleshooting or auditing.
Appendix 2

For pricing intercompany transaction flow with Global Procurement, you need to decide which warehouse organization to use to get the initial cost or use the PO price as the initial cost. The Global POs do not originate from any warehouse, but from the vendor site and we need to arbitrarily select a warehouse that belongs to the procuring OU and always has the cost of all items purchased in this flow.

If you decide to use your master costing organization where the costing organization_id=123 then your costing formula would be:

\[
\text{cst\_cost\_api.get\_item\_cost}(1, \text{INV\_IC\_ORDER\_PUB\_G\_PROC\_LINE}\text{.item\_id}, 123, \text{NULL}, \text{NULL})
\]

Note that the record structure for purchasing is INV\_IC\_ORDER\_PUB\_G\_PROC\_LINE instead of INV\_IC\_ORDER\_PUB\_G\_LINE.

Sourcing cost for master organization can be very tricky in multicurrency environment as the cost in the master costing organization is stored in a single currency while Global PO can be entered in multiple currencies.

You can avoid that by defining initial transfer organization for your Intercompany Global PO flow and then mapping it to your pricing attribute:

\[
\text{cst\_cost\_api.get\_item\_cost}(1, \text{INV\_IC\_ORDER\_PUB\_G\_PROC\_LINE}\text{.item\_id}, \text{XX\_QP\_ATTRIBUTE\_SOURCING\_GET\_PROC\_SHIP\_FROM\_ORG\_ID}(\text{INV\_IC\_ORDER\_PUB\_G\_PROC\_LINE}\text{.ORG\_ID}), \text{NULL}, \text{NULL})
\]

Where XX\_QP\_ATTRIBUTE\_SOURCING\_GET\_PROC\_SHIP\_FROM\_ORG\_ID is a custom package defined as:

```sql
FUNCTION GET_PROC_SHIP_FROM_ORG_ID
(p_ORG_ID IN NUMBER) RETURN NUMBER
IS
x_from_organization_id number;
BEGIN
select distinct
  t_lin.from_organization_id
into x_from_organization_id
from mtl_transaction_flow_headers t_hdr,
  mtl_transaction_flow_lines t_lin
where t_lin.header_id=t_hdr.header_id
  and t_lin.line_number = 1
  and t_hdr.flow_type = 2
  and nvl(t_hdr.start_date, sysdate-1)<sysdate
  and nvl(t_hdr.end_date, sysdate+1)>sysdate
  and t_hdr.start_org_id = p_ORG_ID;
return x_from_organization_id;
Exception
  When others
  Then
    return null;
END GET_PROC_SHIP_FROM_ORG_ID;
```
In some cases, especially for Average Costing organizations you can use the PO price as the initial cost. The In that case the cost mapping formula would be:

INV_IC_ORDER_PUB.G_PROC_LINE.UNIT_PRICE

Now your currency conversion formula would need to change to convert the currency of the PO into the IC transaction currency. The new cost formula would be:

```sql
GL_currency_api.get_rate(
    INV_IC_ORDER_PUB.G_PROC_HDR.CURRENCY_CODE,
    'USD',
    SYSDATE,
    'Corporate')
```

Note that the expression `INV_IC_ORDER_PUB.G_PROC_HDR.CURRENCY_CODE` references `G_PROC_HDR` record structure since the currency is specified at the PO header. Despite that the record structure of `G_PROC_HDR` can be referenced in the User Value String on the line level.

We recommend mapping the Global PO cost (and conversion rate if using PO pricing) into a separate pricing attributes then use an NVL statement in the pricing formula between the PO cost and the sales cost from sales IC flows, or qualified price list to select it. This approach is more transparent, especially for troubleshooting.
Appendix 3

Mapping individual segments of item category in the Intercompany Product Context allows creating separate intercompany markups per product type, i.e. 10% markup for all computers but 15% for laptops.

In this example we will map both segments of the Inventory Item category as separate product attributes. The same approach can be used to map segments of any item category or item catalog.

The first step is to create PL/SQL mapping package. Below is an example of such a package. It will work only if the item category segments are entered in independent value sets prior to creation of category combinations.

```
CREATE OR REPLACE package XX_QP_ATTRIBUTE_SOURCING as
    FUNCTION GET_FAMILY
    (P_INVENTORY_ITEM_ID IN NUMBER,
     P_ORGANIZATION_ID NUMBER ) RETURN VARCHAR2;

    FUNCTION GET_CLASS
    (P_INVENTORY_ITEM_ID IN NUMBER,
     P_ORGANIZATION_ID NUMBER ) RETURN VARCHAR2;

END XX_QP_ATTRIBUTE_SOURCING ;
/
CREATE OR REPLACE package body XX_QP_ATTRIBUTE_SOURCING as
    FUNCTION GET_FAMILY
    (P_INVENTORY_ITEM_ID IN NUMBER,
```
P_ORGANIZATION_ID NUMBER ) RETURN VARCHAR2
IS
x_FAMILY varchar2(200);
BEGIN
    select  ffvv.flex_value_meaning
    into  x_FAMILY
    from mtl_item_categories_v cat, mtl_system_items_b msi, FND_FLEX_VSET_V ffv,
    FND_FLEX_VALUES_VL ffvv
    where
    cat.inventory_item_id=msi.inventory_item_id
    and cat.organization_id=msi.organization_id
    and ffvv.FLEX_VALUE_SET_ID=ffv.FLEX_VALUE_SET_ID
    and ffv.PARENT_VALUE_SET_NAME = 'Item Categories'
    and cat.segment1=ffvv.flex_value
    and msi.inventory_item_id=P_INVENTORY_ITEM_ID
    and cat.organization_id=P_ORGANIZATION_ID;
    return x_FAMILY;
EXCEPTION
    WHEN  others
    THEN
        RETURN substr(sqlerrm,1,200);
END GET_FAMILY;

FUNCTION GET_CLASS
((P_INVENTORY_ITEM_ID IN NUMBER,
  P_ORGANIZATION_ID NUMBER ) RETURN VARCHAR2
IS
x_CLASS varchar2(200);
BEGIN
    select  ffvv.flex_value_meaning
    into  x_CLASS
    from mtl_item_categories_v cat, mtl_system_items_b msi, FND_FLEX_VSET_V ffv,
    FND_FLEX_VALUES_VL ffvv
    where
    cat.inventory_item_id=msi.inventory_item_id
    and cat.organization_id=msi.organization_id
    and ffvv.FLEX_VALUE_SET_ID=ffv.FLEX_VALUE_SET_ID
    and ffv.PARENT_VALUE_SET_NAME = 'Item Class Category'
    and cat.segment2=ffvv.flex_value
    and msi.inventory_item_id=P_INVENTORY_ITEM_ID
    and cat.organization_id=P_ORGANIZATION_ID;
    return x_CLASS;
EXCEPTION
    WHEN  others
    THEN
RETURN substr(sqlerrm,1,200);
END GET_CLASS;
END XX_QP_ATTRIBUTE_SOURCING;

The package allows mapping 2 separate pricing attributes one for product Family and one for product Class.
XX_QP_ATTRIBUTE_SOURCING.GET_FAMILY(INV_IC_ORDER_PUB.G_LINE.INVENTORY_ITEM_ID,INV_IC_ORDER_PUB.G_LINE.SHIP_FROM_ORG_ID)
Both product attributes need to be mapped for IC and INTORG request types.

Run ‘Build Attribute Mapping Rules’ after defining the new attributes.
Appendix 4

If your incompatibility resolution for the List Line Adjustment pricing phase is set to BEST PRICE you may not be able to use it for intercompany modifiers as it can result in the lowest qualifying price being applied to an intercompany transfer instead of the most appropriate one.

In this example the modifier will apply 5% surcharge on all laptops, 4% on all other computers and 2% for all other items only if the incompatibility resolution is set to PRECEDENCE. If it is set to BEST PRICE 2% will be applied for all items including LAPTOPS and COMPUTERS. This is because LAPTOPS and COMPUTERS belong to ALL items and all items has lowest surcharge, therefore the BEST PRICE (or lowest) is the surcharge of 2%.

The incompatibility resolution may on List Line Adjustment may be changed to PRECEDENCE as on the picture below, but it may affect customer pricing which is not our intention.
To avoid this we need to define our separate intercompany pricing phase that includes only the “INV: Batch Processing for Intercompany Transfer Pricing” pricing event.

Check the Freeze Override check box if you are applying freight charges to IR/ISO shipments during Ship Confirm.
Assign the lines of the intercompany modifier to this new phase.