Advanced Pricing Implementation at Mercury Marine (Brunswick)

Sanjay Jain
Fujitsu Consulting
Laura Fleischman
Melissa Neumann
Mercury Marine

Abstract

This paper illustrates key design features used to implement Advanced Pricing to meet complex Pricing requirements at Mercury Marine. Two different Legacy systems were retired. Implementation details of QP integration with ONT and IBE are given. The paper details the List Price design through search based precedence hierarchy; the method used to implement product restrictions through Advanced Pricing; some Advanced Pricing extensions used to overcome seeded functionality limitations; and Modifier Accounting customization.

About Mercury Marine

Mercury Marine is the world's leading manufacturer of recreational marine propulsion engines. A $2.3 billion division of Brunswick Corporation (NYSE: BC), Mercury and its 6,200 employees worldwide provide engines, boats, services and parts for recreational, commercial and government marine applications. Mercury's industry-leading brand portfolio includes Mercury and Mariner outboard engines; Mercury MerCruiser stern drives and inboard engines; Motor Guide trolling motors; Mercury and Teignbridge propellers; MotoTron electronic controls; Mercury inflatable boats; Mercury Smart Craft electronics; and Mercury and Quicksilver parts and oils. Mercury Marine's OptiMax engine was ranked "Highest in Customer Satisfaction with Two-Stroke Outboard Engines Two Years in a Row, Tied in 2007" by J.D. Power and Associates, and Mercury Mercruiser engines were ranked "Highest in Customer Satisfaction with Stern drive Engines, Three Years in a Row."

Implementation Background

Mercury Marine is implementing a four phase program spread over four years called *Business Process Transformation (BPT).* The goal of the Mercury Marine Business Process Transformation is to create an enterprise-wide solution with a common information technology foundation positioning Mercury as a globally integrated business.

The Transformation consists of four "Waves". Each Wave consists of multiple projects. Wave 1 was comprised of projects focused on the fundamental aspects of Mercury data. These projects included: a) building business systems for the China facility, b) building a new data infrastructure to store the company information and c) establishing a Product Lifecycle Management system to manage product data as well as changes to this data.

Wave 2 consisted of projects related to transactional business management processes. This included Oracle Advanced Pricing as part of the "Order to Cash" wave; which is the basis of this paper.

The goal for the team was to create common, end-to-end processes across businesses. This was accomplished by designing globally and initially implementing within the U.S., Canada, Latin America, and Mexico (Juarez). Requirements for all business units, products and organizations for these locations were considered.

The key challenge at Mercury Marine was to design pricing so that all various business areas and customer structures could be taken into consideration. Due to the complexity of the customer base and products Mercury Marine sells, many different approaches had to be taken into account to fulfill Mercury's needs.

List Price Design Challenge

Mercury Marine had two different legacy systems for their Engines and Parts & Accessories (P&A) business units respectively. The Mainframe based PIMS was mainly for P&A and was totally customized. Engines used an older ERP system: IMI ESS. These two systems were managed independently of each other and the Customer orders were entered in both systems since the Product group was different. There were occasions when parts were ordered along with the Engines, and for such scenarios those specific parts and their pricing was also setup in IMI. In addition to these host systems, there was also a custom ebusiness application called Mercnet which had two different sites, for these two broad product groups.

Not only were there two different systems, two different Admin groups managed the pricing for these two different business units. The two business units followed totally different strategies when pricing and therefore have different customer classifications and pricing tiers.

The key differences in the two methods follow:

1. P&A manages their customer base in many more groups, and they used to have Franchise Codes to group the various kinds of customers. Each Franchise code was mapped to a Pricing Bucket. The system provided for maintaining up to 19 prices for the same part in Item master. A map between Franchise Code and Pricing bucket decided the price for a Customer.
2. Engines managed their pricing by a much broader segmentation of customer base called Divisions. They had an MSRP, Dealer Price, Distributor Price and OEM Price.
3. The 2 groups offer and publish different Promotional Programs and have different frequency for Price revision.

The following key decisions were made during O2C implementation which posed a challenge for pricing design:

1. There would be a single Order Entry system and Oracle Order Management and iStore would be implemented to replace IMI, PIMS, and Mercnet.
2. Both P&A and Engines orders would be taken on a single sales order/iStore site.

The decisions were made to enhance customer experience so they would not have to order multiple times for different product groups.

The Business process however was not changing. The Admin groups would still remain different, their pricing strategies would still remain different and they would implement security measures, so that P&A could not make pricing changes to Engines and vice versa.

Design Options Considered

The TCA design established that legacy Franchise code would map to Customer Class and the Divisions (Dealer, Distributor etc.) of IMI would be mapped to Sales Channel. Therefore, the Pricing hierarchy requirements would be represented as follows:

For Engines:

The Pricing would be determined by the Sales Channel unless there is Customer specific Pricing contract.

For P&A:

Pricing would be determined by customer class unless there is a customer specific pricing contract. The number of customer classes was much greater than the Sales Channels.

The following options were available for List Price design:

- Default Price List based on Rules.
  - Pricing Engine is made to point to a specific Price List before Pricing Call.
  - Engine searches for Price in this Price List Only
- Secondary Price List
  - Defaulted Price List can be 'extended' by linking additional Price Lists by secondary relationship.
  - If Price is not found on Primary, the secondary linked to primary is searched for price.
- Precedence based Open search
  - Price Lists can be qualified by Customer Attributes with a pre-assigned but updateable precedence value
  - Engine evaluates all Qualifier Attribute values and fetches the Price Line with least precedence value
  - Since the engine does not point to a Price List to start, there is no primary and hence Secondary functionality does not work.

Behavior of iStore and Order Management vis-à-vis above options:

Order Management has the best integration with Advanced Pricing and therefore any kind of scenario can be handled for List Price when the Pricing call is made for ONT application.

Defaulting Rules can be based on any Line or Order attribute and each order line can be made to default to a different price list based on attributes using sophisticated defaulting rules. Further, search based Pricing can be setup, which would search the appropriate Price based on Qualifiers.

The best part is Defaulting and Search based Pricing can be setup complementing each other- so that the Defaulting Rules could be setup as far as possible and if the Price is not found on the default Price List, the Pricing engine would try to get the Price using an open search.

iStore to pricing integration, however, is not so versatile and has limited options:

- Point the login session to a specific Price List
  - The only seeded option for this usage: default from Customer Account.
  - Through Java layer customization, default Price List can be any Price List.
  - Secondary Functionality works once Primary is established.
  - Defaulting is possible solely on Customer Attributes. (Not Item based).

- Do not point login session to specific Price List.
  - Use QP Pricing Engine to get Price.

Even though defaulting is somehow achieved, the Default List must be used for the whole login session- so one cannot have the control to default to a different Price List on each Line. Furthermore, since item attributes cannot be 'seen' by the login session, defaulting based on Product attributes was ruled out.

Once the session is pointed to a specific price list, it cannot be made to search for the Price using open search. In effect, it is either Defaulting or Search. The two methods cannot work together (in contrast to Order Management, which allows Defaulting and Search to work together).

List Price Design for Mercury Marine

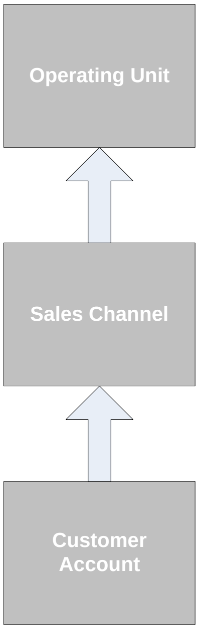
Based on the above stated requirements and limitations, it was concluded:

1. Product Based Defaulting Rule for Price List could be used since P&A and Engines would have different Price Lists and different sets of Qualifiers. Such defaulting rule would only work for Order Management and not for iStore.
2. Defaulting based on One Product Group was considered (since iStore session cannot see Product Attributes for Price List defaulting) , but then the session would be stuck at this Price List and would not search for a price for the other Product Group. This would also only work in Order Management and not in iStore.
3. The only option which would work, both for iStore and Order Management would be the Precedence based search.
4. Defaulting Rules for Price List would be setup only for scenarios when the Orders would be taken in Order Management and not through iStore.

Engines Pricing Structure

Engines products have 3-tier pricing. The three tiers will be:

Customer Account
Customer Sales Channel (or a group of Sales Channels)
Operating Unit



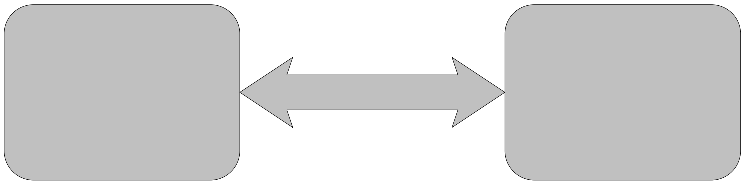
Customer Account Level: This represents the scenario when a specific Customer has its own negotiated base Price. A separate Price List will be maintained for each such customer and shall be qualified by the specific customer account. These price lists could have price lines listing each item or could be based on formulas depending on uplift from standard cost.

Customer Sales Channel Level: A Sales Channel; e.g. Dealers, get a different base price for Mercruiser products than the other. A separate Price List will be maintained for each such scenario and will be qualified by the specific Sales Channel.

Operating Unit Level: There will be one base price list for each OU. If a customer does not have a customer level or Sales Channel level pricing, it will, by default have the Operating Unit base price.

P&A Pricing Structure

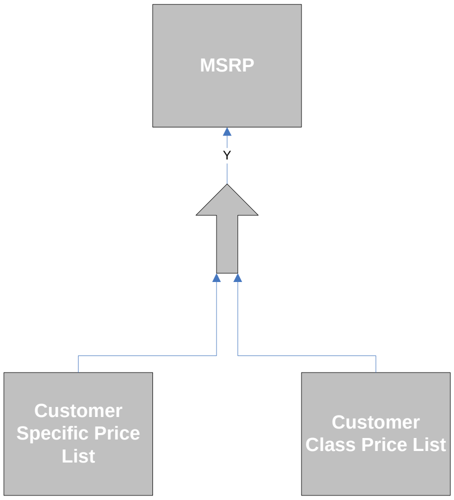
P&A will have one Price List for each Pricing Bucket. There were up to 19 Pricing Buckets for each Part maintained within the item master. Of these, only six in the USA and three in Latin America were migrated to Oracle. As and when the need arises, a new Bucket would be created or an existing one dropped. Accordingly, if a new Bucket is created, a new Price List would be created in Oracle for that Price Bucket.



The Pricing Buckets broadly represent the type of Pricing a Customer gets due to its class (Franchise Code in legacy world). Examples include:
Dealer
Distributor
OEM
MSRP

If an item is not available for a Customer Class price list, it can still be ordered at MSRP. The concept of selling at MSRP was introduced with this implementation. In the previous systems, MSRP was only a published price and if a product was not available to a customer as per system established price, the customer was directed to procure the product (typically Parts and Accessories) from authorized dealers. With Oracle implementation, the product would be available from Mercury. This was termed at ‘Soft Restriction’ and resulted in bringing additional business to Mercury.

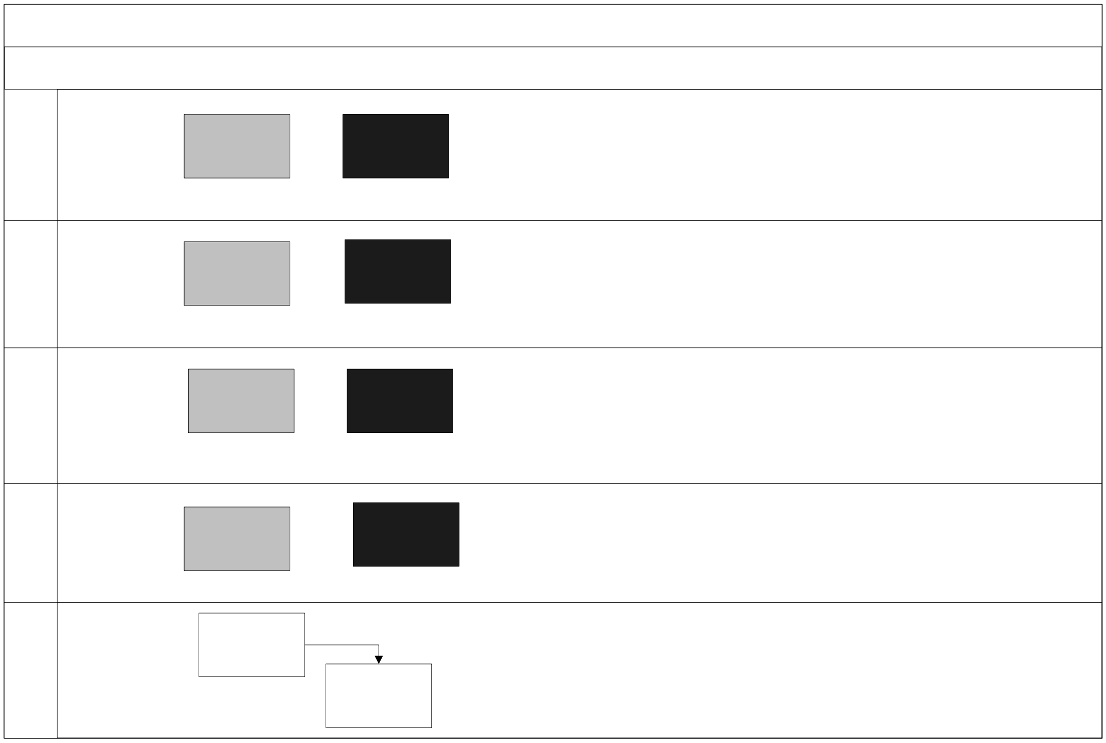
The following flow depicts the most typical flow:



There are exceptions to this typical structure and the system is required to provide 3-tier hierarchy:

Customer Account → Customer Class → Operating Unit.

Assimilating the requirements for both P&A and Engines, the following design was implemented:



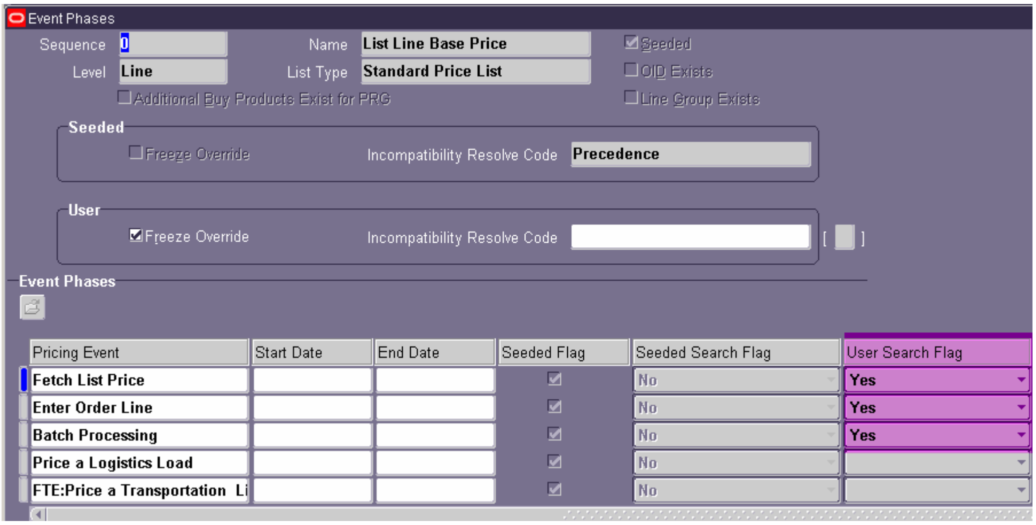
The design relies on Precedence. A custom context of Pricing Level is configured with the following five Qualifier Attributes:

Qualifier Attribute	Precedence Value
Customer Account	20
Customer Group	30
Customer Class	40
Sales Channel	50
Operating Unit	60

Each of the Price Lists is qualified by at least one of the above qualifiers. The Pricing Engine searches for the most favored Price by looking at the least precedence value.
The design provides for controlling this hierarchy at the Operating unit level and so there is different Pricing data for USA, Latin America and China. Security features are built in at Responsibility Level, to control access to Pricing Entities across Operating units and roles.

Although the above mentioned default precedence values are setup, they can be manipulated to accommodate special scenarios which will not necessarily be served by this model.

The precedence based search is achieved by setting up Event Phases:



Event Phases

Sequence	0	Name	List Line Base Price	☑ Seeded
Level	Line	List Type	Standard Price List	□ OID Exists
	□ Additional Buy Products Exist for PRG			□ Line Group Exists

Seeded

□ Freeze Override   Incompatibility Resolve Code: Precedence

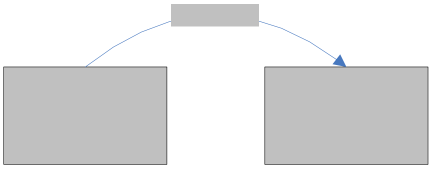
User

☑ Freeze Override   Incompatibility Resolve Code:

Event Phases

Pricing Event	Start Date	End Date	Seeded Flag	Seeded Search Flag	User Search Flag
Fetch List Price			☑	No	Yes
Enter Order Line			☑	No	Yes
Batch Processing			☑	No	Yes
Price a Logistics Load			☑	No	
FTE:Price a Transportation Li			☑	No	

MSRP: An MSRP Price List will be maintained for each OU. This is required for iStore Catalogue and various Pricing interfaces which publish the Pricing to the external world (Price Books etc.). P&A MSRP Price List will be primary. Engines MSRP list will be secondary to this Price List.



Product Restrictions

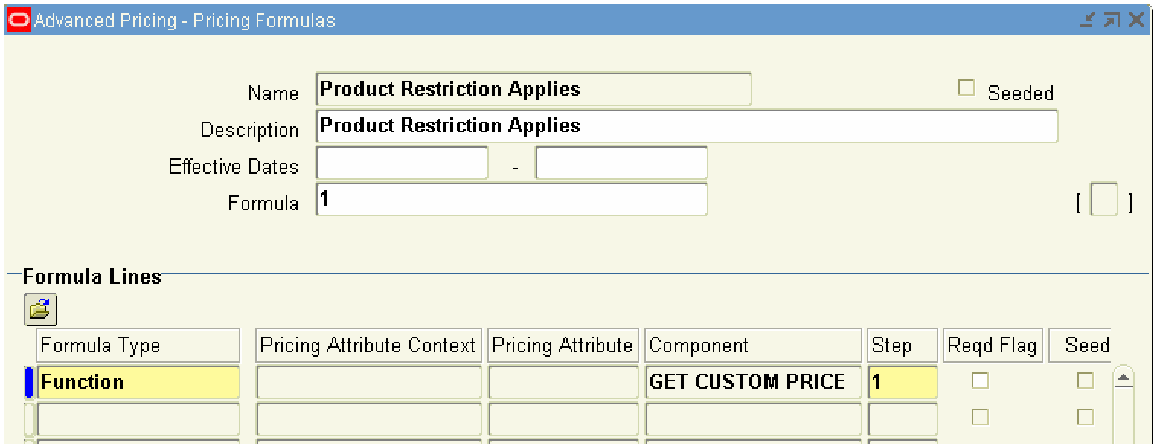
Mercury Marine product restrictions are a complex matrix as they existed in Legacy Systems. One of the major requirements of Mercury business was to be able to enforce Product Restrictions based on certain rules specified and maintained by the business. Product Restrictions is a Mercury Marine term to specify rules about who can order what item. The Rules are primarily defined in terms of Customer Class and Product Line, but can also be defined based on other Customer and Product Attributes (which could be as specific as Customer ID and Product ID).

Product Restriction Rules are based on a number of Customer and Product Attributes and they do not neatly map to the Price List structure. Following are some examples of the restriction rules:

> Mercury has a service classification of “PNA DTS Restriction”. This group of customers (not an attribute on the customer master) is not allowed to order parts and accessories that are used to rig a digital throttle and shift system.
>
> A Customer Class of Propeller Repair Station that is not allowed to purchase any other product from Mercury besides propellers and parts that are associated with repairing these items.

The following options were considered in implementing Product Restrictions:

1. *By not pricing through Price Lists*. This option is not a favored option because pricing rules and restriction rules are totally disjointed and trying to map the two would pose an administrative nightmare. Further, user would never know if a Price Not Found error is due to restriction or due to a data entry error. So, attempt to enforce restriction by Not Pricing was discarded
2. *By implementing Rules Based OM Processing Constraints*. This option was to maintain all restriction rules in some custom matrix, and then use Processing constraint functionality and configure create constraints based on check against this table. This would work fine in Order Management, but for iStore, this was not considered the best approach. The constraint would fire, but only after the Order is submitted and hits the Order Management application. This was considered too late and the iStore user would still see a valid price in catalogue and cart and would learn about the restriction only after the submit button was pressed.
3. *By Conditional Null Pricing*. The final option considered was through Null Pricing. Hard Restriction is achieved using a custom formula referencing a function in Get_Custom_Price. If a constraint condition exists the pricing engine will return a NULL price and user will get an explicit message about restriction condition. NULL Pricing shall be achieved by using the dynamic formula on the Master Price List Line for ALL Items and giving that line rock bottom precedence. The detailed message will only be available in the Order Management application. IStore users will receive a generic message through Java layer customization.



Advanced Pricing - Pricing Formulas

Name: Product Restriction Applies □ Seeded

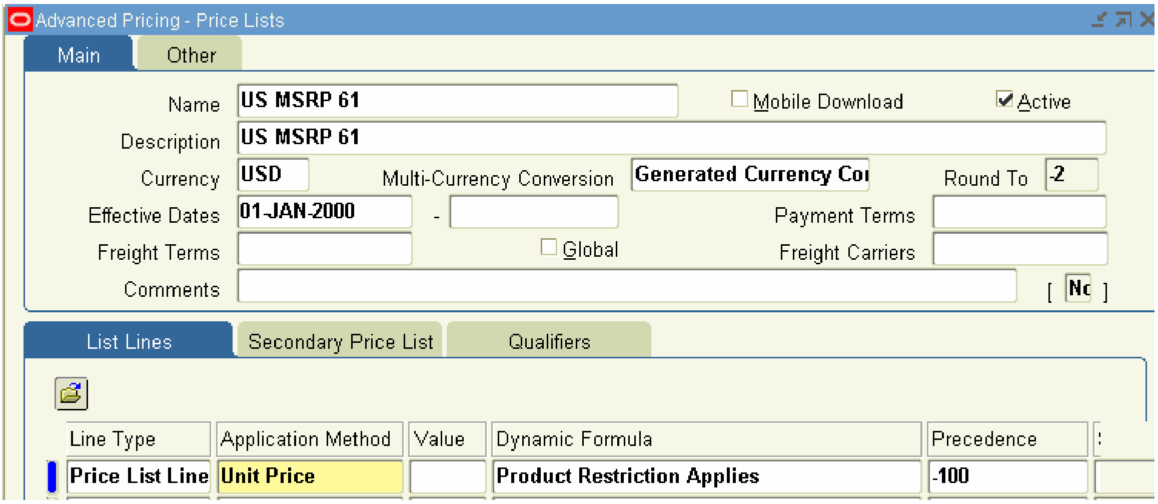
Description: Product Restriction Applies

Effective Dates: -

Formula: 1

Formula Lines

Formula Type	Pricing Attribute Context	Pricing Attribute	Component	Step	Reqd Flag	Seed
Function			GET CUSTOM PRICE	1	□	□
					□	□



Advanced Pricing - Price Lists

Name: US MSRP 61 □ Mobile Download ☑ Active

Description: US MSRP 61

Currency: USD Multi-Currency Conversion: Generated Currency Coı Round To: -2

Effective Dates: 01-JAN-2000 - Payment Terms:

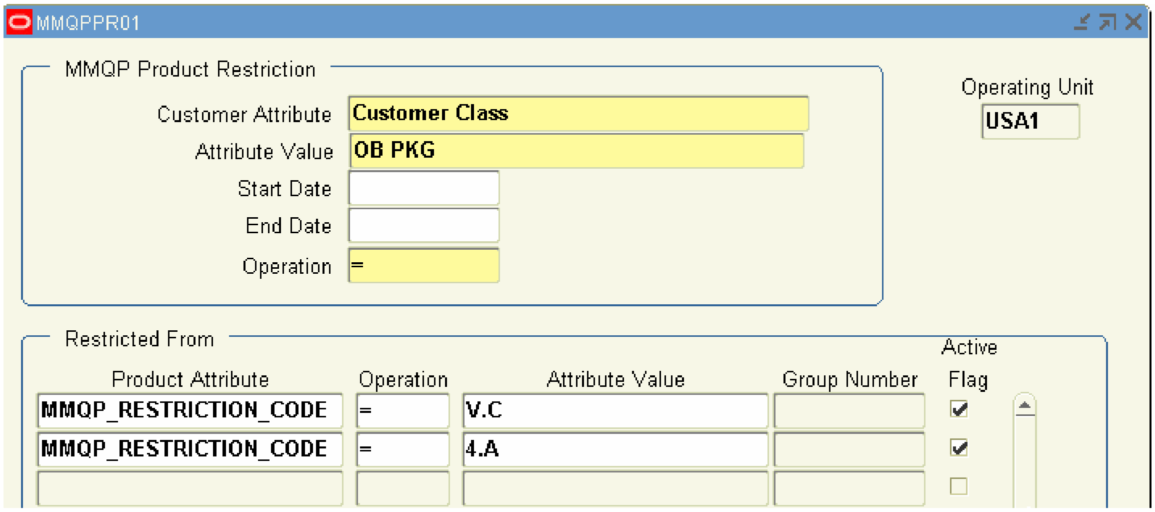
Freight Terms: □ Global Freight Carriers:

Comments: [Nc]

Line Type	Application Method	Value	Dynamic Formula	Precedence
Price List Line	Unit Price		Product Restriction Applies	-100

Advantages of implemented design

Extremely User friendly: User maintains the Restrictions Matrix using Customer and Product Attributes which are easily understood by the business.



MMQPPR01

MMQP Product Restriction

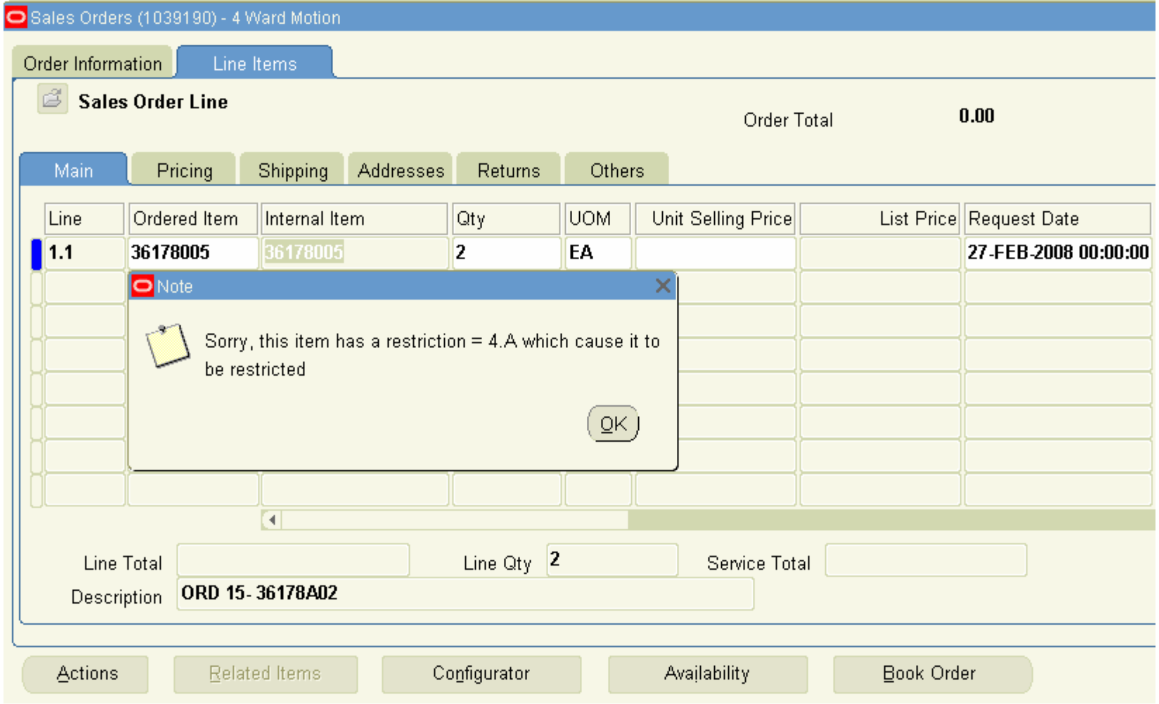
Customer Attribute	Customer Class
Attribute Value	OB PKG
Start Date	
End Date	
Operation	=

Operating Unit: USA1

Restricted From

Product Attribute	Operation	Attribute Value	Group Number	Active Flag
MMQP_RESTRICTION_CODE	=	V.C		☑
MMQP_RESTRICTION_CODE	=	4.A		☑
				□

Identical Behavior in iStore and Order Management: The Pricing engine would return a NULL price using a Pricing Formula and the timing of the behavior would be right at the first pricing call in the Fetch List Price phase.



Sales Orders (1039190) - 4 Ward Motion

Sales Order Line

Order Total 0.00

Line	Ordered Item	Internal Item	Qty	UOM	Unit Selling Price	List Price	Request Date
1.1	36178005	36178005	2	EA			27-FEB-2008 00:00:00

Note

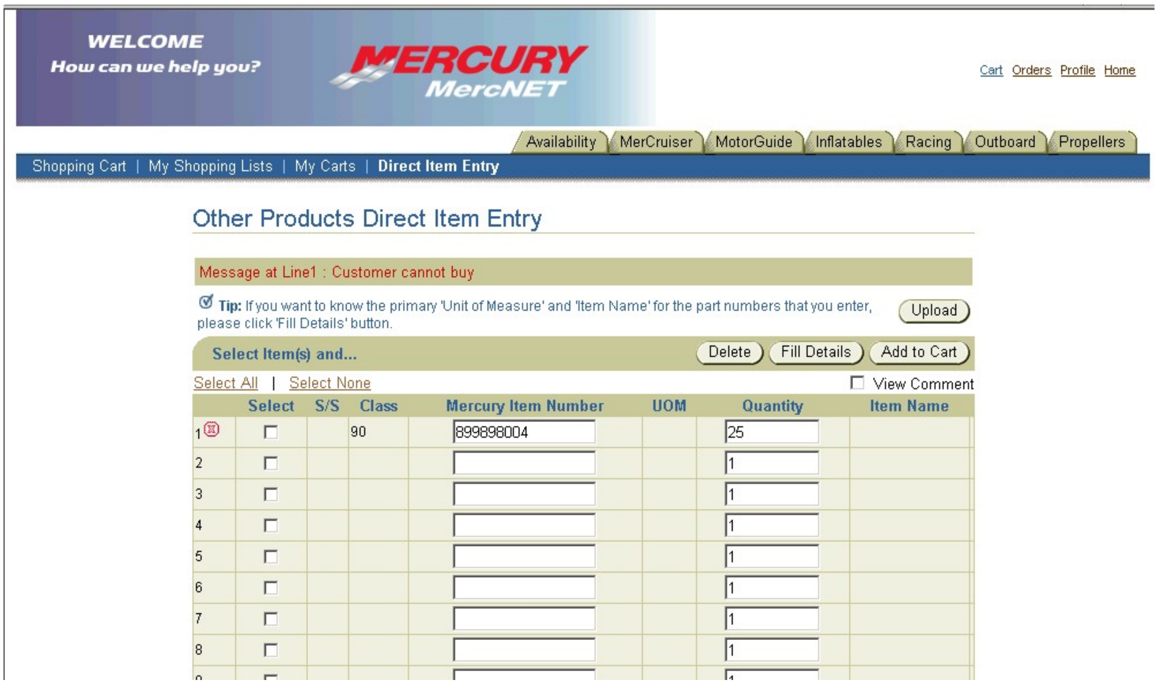
Sorry, this item has a restriction = 4.A which cause it to be restricted

Line Total

Line Qty 2

Service Total

Description ORD 15- 36178A02



Other Products Direct Item Entry

Message at Line1 : Customer cannot buy

Tip: If you want to know the primary 'Unit of Measure' and 'Item Name' for the part numbers that you enter, please click 'Fill Details' button.

	Select	S/S	Class	Mercury Item Number	UOM	Quantity	Item Name
1 ☒	□		90	899898004		25	
2	□					1	
3	□					1	
4	□					1	
5	□					1	
6	□					1	
7	□					1	
8	□					1	

Message for the User informing reason for failure: The Pricing Formula code would also throw a custom message informing the user for the reason of failure. This would only occur in Order Management because visibility is required only to Internal Customer service reps.

Advanced Pricing Extensions

This section covers some extensions which were implemented to meet some very unique scenarios. As mentioned earlier, the Order Process was merged from P&A and Engines orders to mixed Orders within Oracle. This posed fresh challenges to design Promotions and Discounts. P&A and Engines both published their own separate programs which were based on Order Totals or Order Quantities. Since there would now be the same order containing all kinds of Products, Order Quantity or Amount would not work. Several of the Modifiers therefore needed to be defined at 'Group of Lines' level.

Here are some scenarios which were met by defining extensions:

1. *Promotions based on a multitude of Categories and their Category Value* in a specific segment. Examples include:
   Engine Horsepower
   Product Line
   Brand
   Steering

Seeded Functionality Limitation: Seeded Item Category Attribute only works for the Category set which is set up as a default for the functional area of Order Management. Also, a specific segment cannot be used as a pricing attribute.

Extension:

A custom function was written, which would take as input, the Category Set Name, Segment Number and the Item Id and would return the Category segment value. This single function was used to map several Pricing Attributes using appropriate parameters.

2. *One Price List based on the other*: Several Mercury Pricing rules are based on relative pricing or cost. For example, Dealer Price was a specific Product Line that is x% less than MSRP. Employee Price is 5% uplift on cost. Such scenarios were handled using Formulas and again two generic functions were written:
   Get_cost and get_price_from_price_list. Get_price_from_price_list, takes the item_id, Price List Name and Pricing Date as input and returns the Price from that Price List. The function was repeatedly used to map custom pricing attributes for each reference Price List and then those attributes were used on Formulae. This provided a powerful dynamic structure whereby user would need to maintain only those prices which are key to the rest of the prices.

3. *Discrete Customer and Item Groups*: While Customer and Product Attributes and Categories would normally suffice to define sufficient product/pricing/qualifier attributes to setup all pricing and promotions; there are scenarios when it is needed to maintain our own Customer and Item Groups. Examples of such scenarios are:

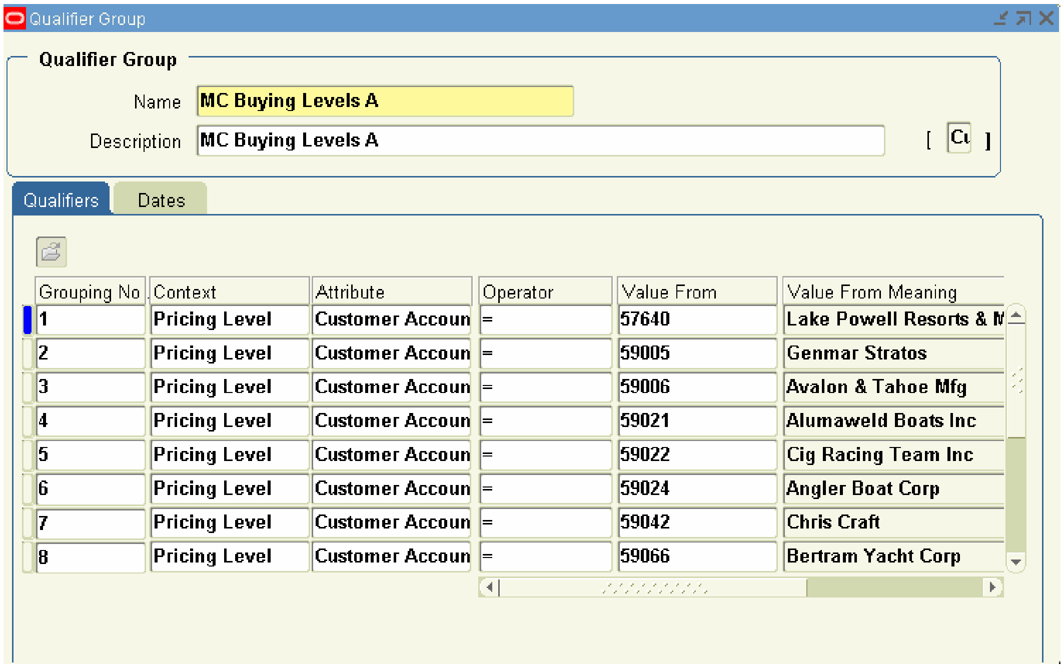
   Customers who achieve certain Service Level in a Sales year
   Customers based on MM Purchases in prior year
   Customers who sign-up for a Promotional program

   Specific Spark Plug Items on Clearance
   Specific Batteries offered in a Promotion

   Seeded functionality provides for maintaining Qualifier Groups, but any change made to the Qualifier Group is not cascaded to the Pricing entities where they are used.
   A promotion based on a group of items, cannot be defined unless there is a common attribute/category.

   To meet the requirement of maintaining discrete Customer and Item Groups, custom attributes are defined called Customer Group and Item Group.

   The Qualifier Group table structure was used to define both Item and Customer Groups and the mapping function would return the Group Name(s) that the Ordered Item and Customer exist in.



Qualifier Group

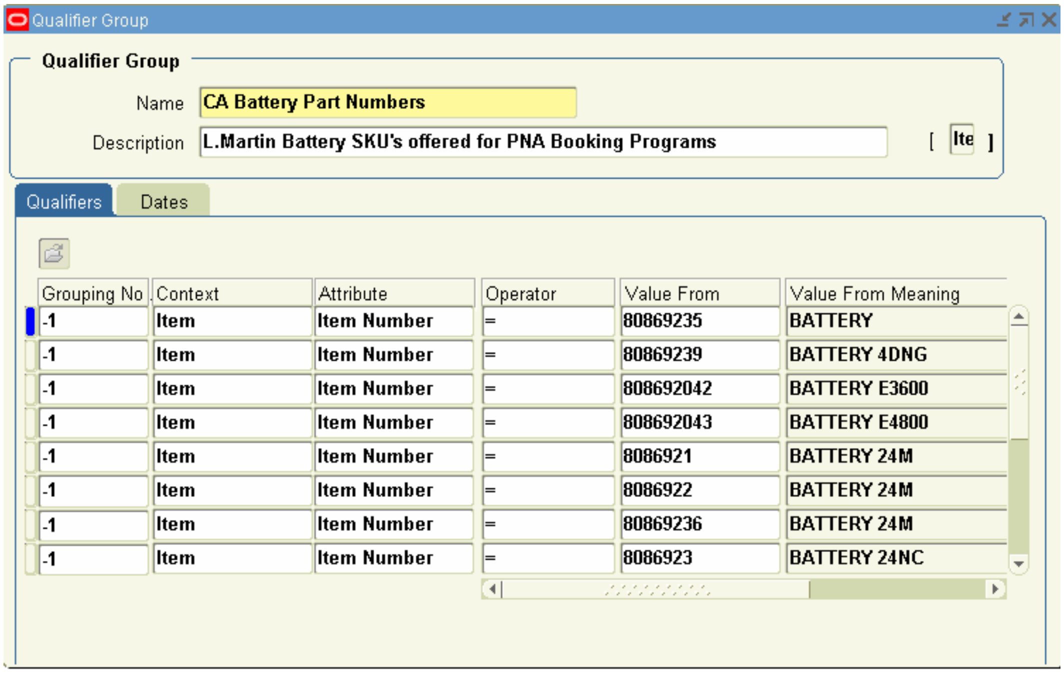
Qualifier Group

Name: MC Buying Levels A

Description: MC Buying Levels A

[Cu]

Grouping No	Context	Attribute	Operator	Value From	Value From Meaning
1	Pricing Level	Customer Accoun	=	57640	Lake Powell Resorts & M
2	Pricing Level	Customer Accoun	=	59005	Genmar Stratos
3	Pricing Level	Customer Accoun	=	59006	Avalon & Tahoe Mfg
4	Pricing Level	Customer Accoun	=	59021	Alumaweld Boats Inc
5	Pricing Level	Customer Accoun	=	59022	Cig Racing Team Inc
6	Pricing Level	Customer Accoun	=	59024	Angler Boat Corp
7	Pricing Level	Customer Accoun	=	59042	Chris Craft
8	Pricing Level	Customer Accoun	=	59066	Bertram Yacht Corp



Qualifier Group

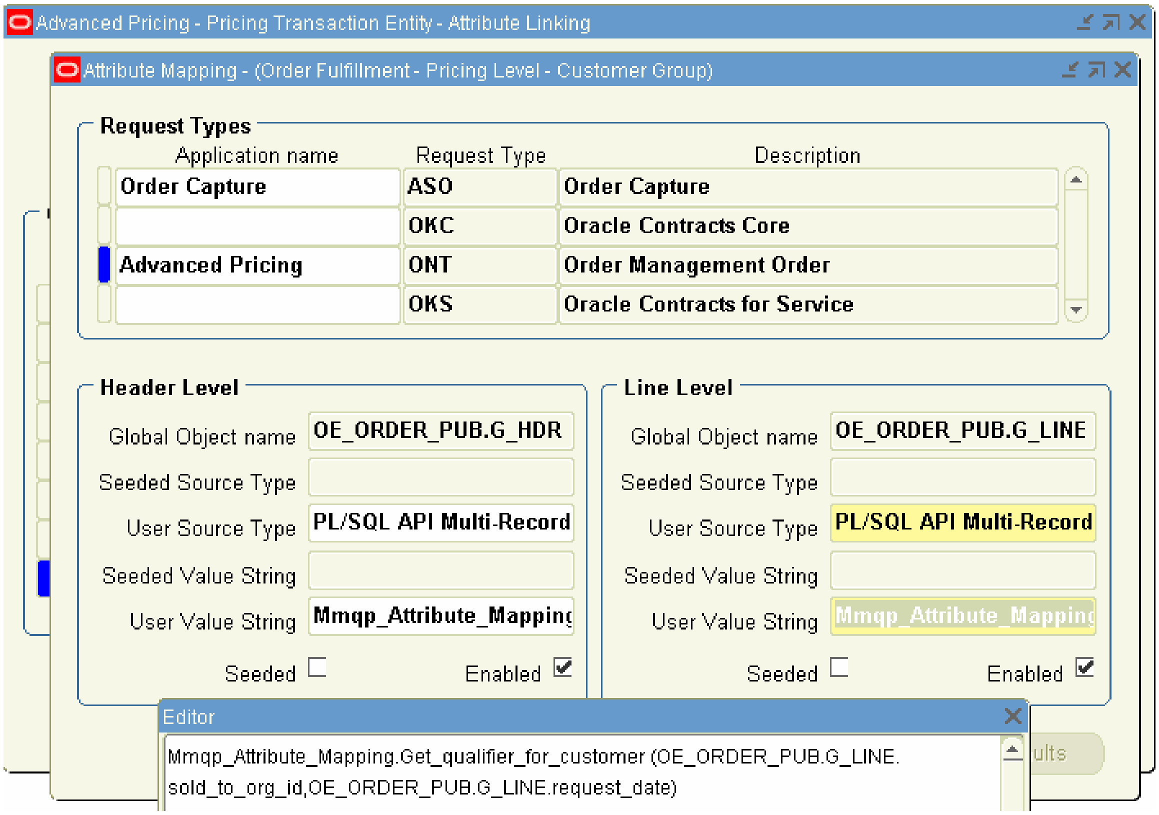
Qualifier Group

Name: CA Battery Part Numbers

Description: L.Martin Battery SKU's offered for PNA Booking Programs [Ite]

Grouping No	Context	Attribute	Operator	Value From	Value From Meaning
-1	Item	Item Number	=	80869235	BATTERY
-1	Item	Item Number	=	80869239	BATTERY 4DNG
-1	Item	Item Number	=	808692042	BATTERY E3600
-1	Item	Item Number	=	808692043	BATTERY E4800
-1	Item	Item Number	=	8086921	BATTERY 24M
-1	Item	Item Number	=	8086922	BATTERY 24M
-1	Item	Item Number	=	80869236	BATTERY 24M
-1	Item	Item Number	=	8086923	BATTERY 24NC

The function would return multiple values and the mapping method of Multi Record PL/SQL is used



Advanced Pricing - Pricing Transaction Entity - Attribute Linking

Attribute Mapping - (Order Fulfillment - Pricing Level - Customer Group)

Request Types

Application name	Request Type	Description
Order Capture	ASO	Order Capture
	OKC	Oracle Contracts Core
Advanced Pricing	ONT	Order Management Order
	OKS	Oracle Contracts for Service

Header Level

Global Object name: OE_ORDER_PUB.G_HDR

Seeded Source Type:

User Source Type: PL/SQL API Multi-Record

Seeded Value String:

User Value String: Mmqp_Attribute_Mappinç

Seeded □ Enabled ☑

Line Level

Global Object name: OE_ORDER_PUB.G_LINE

Seeded Source Type:

User Source Type: PL/SQL API Multi-Record

Seeded Value String:

User Value String: Mmqp_Attribute_Mappinç

Seeded □ Enabled ☑

Editor

```
Mmqp_Attribute_Mapping.Get_qualifier_for_customer (OE_ORDER_PUB.G_LINE.
sold_to_org_id,OE_ORDER_PUB.G_LINE.request_date)
```

4. *Promotions based on UOM different from Pricing UOM*

For Promotions based on Qty break point in UOM different than pricing UOM, custom Pricing Attributes are used which will keep the running total of the Qty in promotion UOM for the specified Item Group. This is added to the Modifier Line with an applicable break point value. This is required to overcome the limitation within seeded functionality which the modifier lines apply only if the Modifier Line UOM is the same as Pricing UOM.

For example, specific Oil products are ordered by Eaches, but a discount applies when the total quantity on the Group of Lines for that kind of Product totals 50 Cases.

Modifier Accounting

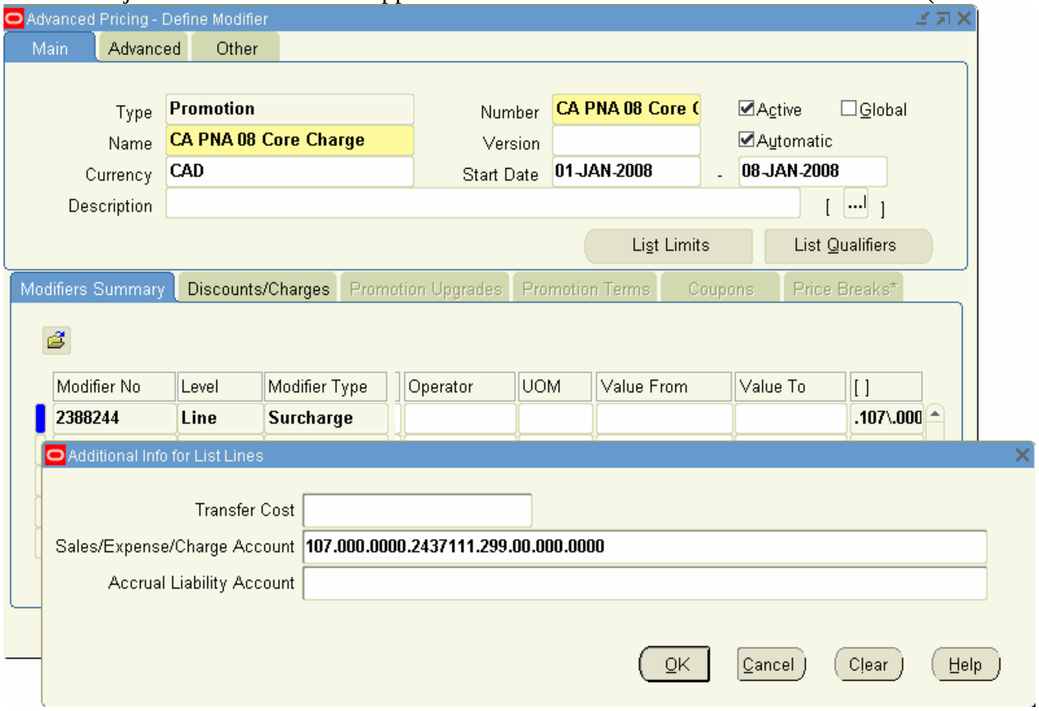
Mercury offers a multitude of Promotion Programs.

To track the promotions transactions, Mercury is required to post these Adjustments to pre-defined GL Accounts which vary by each promotion.

There is no seeded feature within OM/QP/AR Applications to address these requirements. Trade Management Application provides this functionality, but it was decided not to implement this Application in this phase of the Project.

As per seeded functionality, all Off-Invoice adjustments are interfaced to AR, with the same Revenue account as the Order Line to which the adjustment applies. A custom routine was written to identify all such adjustment lines and update the distribution account value on this. Following are the Rules for this update:

1. Off Invoice Adjustments are identified by adjustments where accrual flag is 'No'
2. Adjustment ID is interfaced to AR in column Interface_Line_Attribute11
3. Sales Adjustment Account appears in a DFF on Modifier Line (Attribute2).



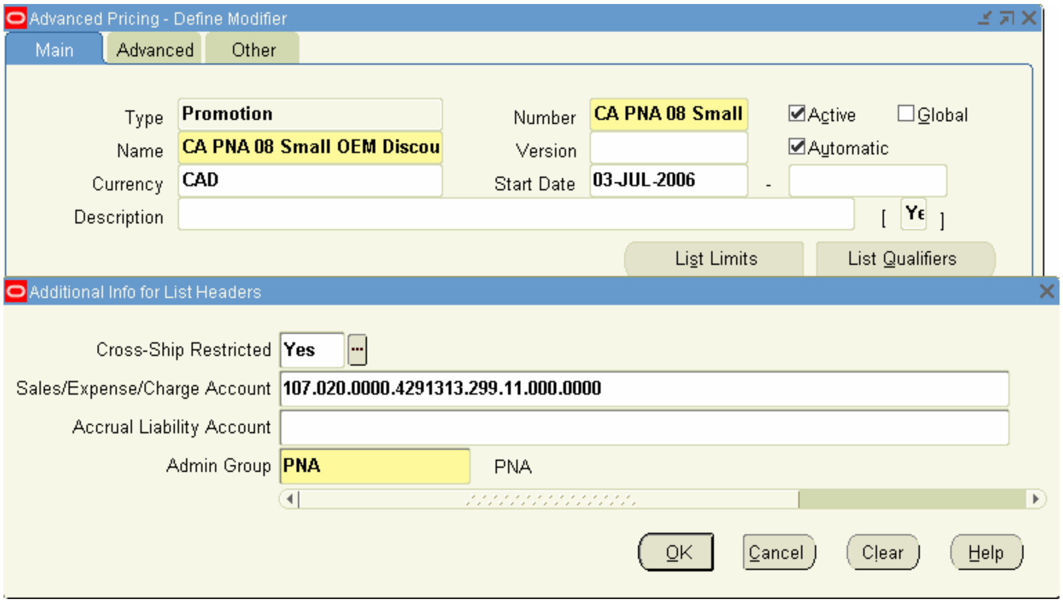
Type: Promotion
Name: CA PNA 08 Core Charge
Currency: CAD
Description:
Number: CA PNA 08 Core (
Version:
Start Date: 01-JAN-2008 - 08-JAN-2008
☑ Active □ Global
☑ Automatic

Modifier No	Level	Modifier Type	Operator	UOM	Value From	Value To	[ ]
2388244	Line	Surcharge					.107\.000

Transfer Cost:
Sales/Expense/Charge Account: 107.000.0000.2437111.299.00.000.0000
Accrual Liability Account:

If an Account Value is not found on these fields or if the Account there is INVALID, look at the corresponding Modifier Header where:

Attribute3= Sales/Expense/Charge Account


Advanced Pricing - Define Modifier
Main
Advanced
Other
List Limits
List Qualifiers
Additional Info for List Headers
OK
Cancel
Clear
Help

Type: Promotion

Name: CA PNA 08 Small OEM Discou

Currency: CAD

Description:

Number: CA PNA 08 Small

Version:

Start Date: 03-JUL-2006 -

☑ Active ☐ Global

☑ Automatic

[Ye]

Cross-Ship Restricted: Yes

Sales/Expense/Charge Account: 107.020.0000.4291313.299.11.000.0000

Accrual Liability Account:

Admin Group: PNA PNA

If an Account Value is not found on the Header or if the Account there is INVALID, look at the custom Profile Options at Org Level created for this purpose.

Conclusion

Mercury Marine implementation of Advanced Pricing illustrates the extreme capabilities of this product when exploited. The Product provides ability to integrate with multiple applications using the same data set with identical pricing on transactions. There are several ways to configure List Price and each implementation probably requires a unique setup and design. In this instance, iStore integration was the main factor in deciding the design to go for precedence based search. Product Restrictions, very unique promotions and Modifier Accounting requirements are very effectively handled in this design.

Mercury Marine is live since April 2007 and we have been supporting this system since. In our experience, the design has been largely successful. After analyzing the reported Support issues over the past year, we realize that this design is heavily based on data integrity. We have used several Pricing Attributes which rely on strict discipline on the Data Administration team to maintain the attributes on items and customers. Most of the reported issues were traced back to wrong data for the source attribute.

In conclusion, we are very proud of this successful implementation and reiterate that strict data integrity is the key to success for any Attributes based design.