***Business Reason***

Small container competitive bid scenarios for existing customers originally specified that floor price guardrail would be set at the max of competitive bid or cost. Floor price guardrail needs to be revised to select the max of competitive bid or new business floor pricing.

***Task Requirements***

Floor price on competitive bid should reflect new business floor pricing for that container configuration.

* For small container existing customer competitive bid scenario, update floor price guardrail (referred to as “base price” in BML) to reflect the formula found in the attached patch file
* Rule should reflect the table below:

Competitive\_Bid\_Markups

|  |  |
| --- | --- |
| 1. Base Price | Min(Max(New Business Floor, Competitive Bid \* Competitive Base Markup),Current Price) |
| 1. Target Price | Base Price (A) + PriceGap\*TargetRetainedPercent |
| 1. Stretch Price | Base Price (A) + PriceGap\*StretchRetainedPercent |

*Price Gap = Current Price – Base Price (A)*

**Old Calculation**

Price\_base = cr\_competitive\_bid \* min(revenue, competitive\_bid\_amt\*comp\_bid\_markup)

**New Calculation**

Price\_base = cr\_competitive\_bid \* min(revenue, max(cost\_to\_serve\_month/(1-base\_margin\_new), competitive\_bid\_amt \* comp\_bid\_markup))

**Variables**

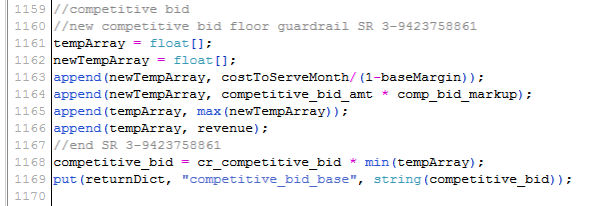
Cr\_competitive\_bid = an indicator [0,1]. It is 1 when we have a competitive bid rollback.

Competitive\_bid\_amt is a float variavle associated with competitiveBidAmount\_sc on the configurator.

Com\_bid\_markup is the value returned from the CompetitiveBid\_Markup data table.

***Build Plan***

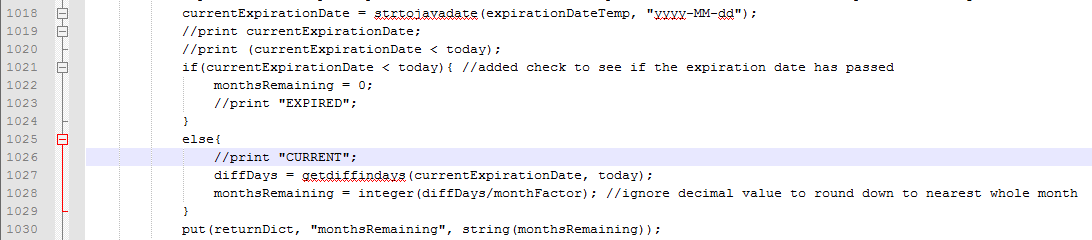
1. The floor (totalBasePrice\_line) is calculated in the “Calculate Guardrails” util.
2. See line 1159 for where the competitive bid is calculated.
3. I added the following code:



1. We use “newTempArray” to get the maximum between cost\_to\_serve\_month/(1-nbase\_margin\_new) and competitive\_bid\_amt\*comp\_bid\_markup.
2. We use “tempArray” to get the minimum between revenue and the max of “newTempArray.”

Additional Change: Expired Contracts

1. “monthsRemaining” is used to determine how many months is left in a contract, the code was not considering contracts that had expired and therefore returned the number of months since contract expiration.
2. Code added to “caculateGuardrails” on lines 1021-1029:



1. If contract is expired, “monthsRemaining” is set to “0”.
2. Otherwise, “monthsRemaining” is set to the difference in days between today and the contract expiration date divided by “monthFactor”.

Unit Test

1. To create a competitive bid rollback scenario:
   1. Create a quote for an existing customer.
      1. Ex. A46 – 4753 – account 233-13840
      2. Sample quote A147386
   2. Under View/Change Existing Services, click “Next”.
   3. Under Sales Activity, select “Rate adjustment.”
   4. Under Price Adjustment Reason, select “Rollback due to Competitive Bid.”
   5. Enter a Competitive Bid Amount.
   6. Save.

***Admin Guide Changes***

The competitive bid floor is calculated in the “Calculate Guardrails” util. See line 1159 or search for “3-942375861” in the comments to find the code snippet. The price is now cr\_competitive\_bid \* min(revenue, max(cost\_to\_serve\_month/(1-base\_margin\_new), competitive\_bid\_amt \* comp\_bid\_markup)). The max is handled in the “newTempArray” and the min is handled in the “tempArray.”

***User Guide Changes***

None.