GeneXus by Globant

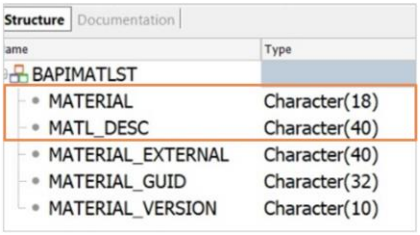
Integrating the KB to SAP ERP

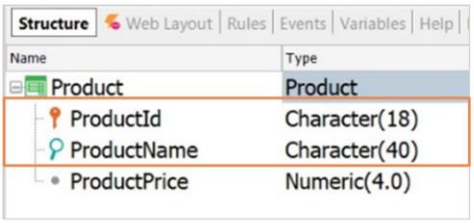
Running the GetList method

We have previously established the connection to SAP ERP and invoked the GetList method of the Materials BAPI to receive the list of materials.

Our goal now is to run through this list and put it in the Product table of our application.

Running the GetList method



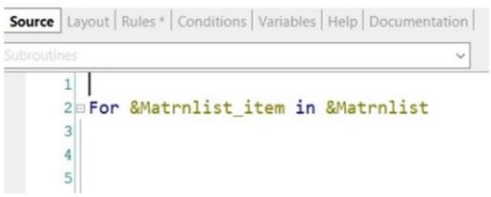


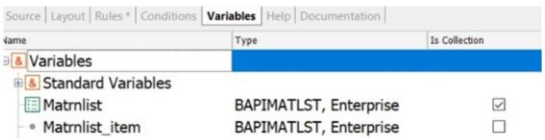
First, note that in the definition of our Product transaction the Id is autonumbered, and the name is Character of length 20.

However, in the BAPIMATLIST SDT structure we can see that the number associated with the material is defined as a Character of length 18 and its description as Character of length 40.

So we modify the structure of our transaction to be compatible with these definitions. OK. We must now define the process that runs through the materials list received from the ERP and puts it in the product table.

Running the GetList method





We create a procedure named UserUpdateProducts.

This procedure must receive the list of materials returned by the execution of the GetList method, so we define the &MatRnList variable of the BapiMatList data type, and set it as a collection.

We go to the Rules tab and define the corresponding Parm rule.

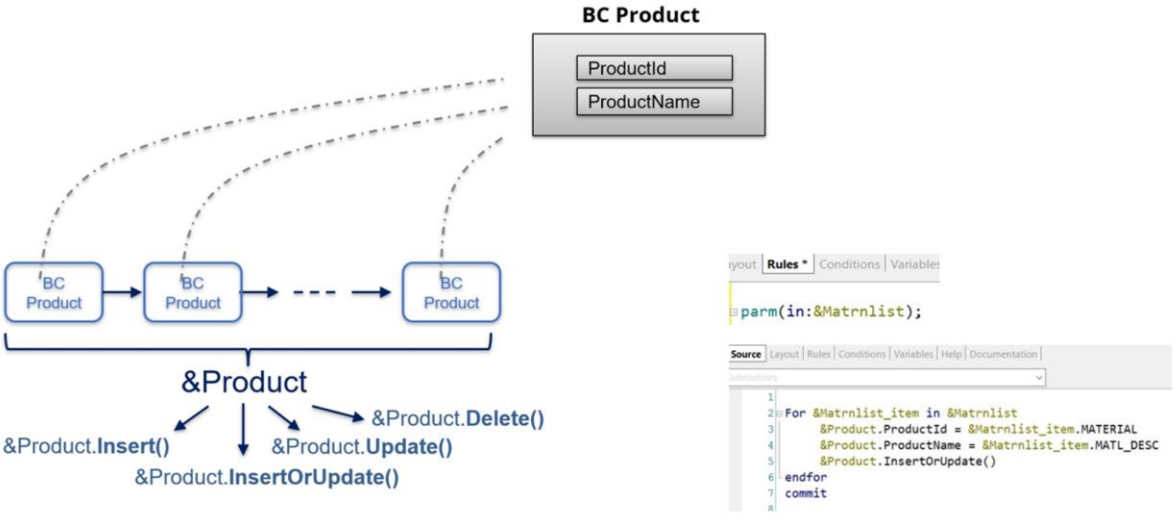
We need to run through this collection of materials, so we must define another variable of BapiMatList type but this time to represent an element of the collection, so we define it as simple and not as a collection.

OK. We now go to the source to run through the list of materials:

And for each of these elements, we must insert it as a new product in our table if it does not exist, or update it if it already exists.

So we declare our Product transaction as a Business Component, and use the InsertOrUpdate method to process the records.

Business Component: Product



Remember that Business Component is a transaction property that allows triggering its logic outside of the transaction.

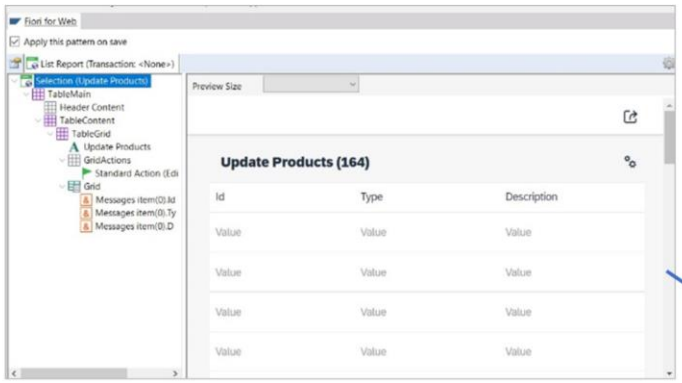
The methods we are looking at apply to both simple variables and Business Component collections. In particular, the InsertOrUpdate method has a specific behavior, because it first tries to insert the record, and if it fails due to a duplicate key, it retrieves and updates it.

This is the behavior we need to meet this requirement since our goal is to always keep the product table up to date, inserting new products or updating the existing ones.

So we return to the procedure, and define the &Product variable, Business Component.

In the source, we complete the code. After assigning the values for the Id and product name, we apply the InsertOrUpdate method. We close the for and declare the commit, which is necessary when working with Business Components.

Running the GetList method



Start event

```
SAPMaterialGetList(&Matrnlist, &Messages)

if &Messages.Count = 0
     UpdateMyProducts(&Matrnlist)
     msg("Products were updated")
else
     msg("Errors in the execution")
endif
```

OK. At this point, we need an object that triggers these processes. So we create a web panel named UpdateUserProducts and apply the Fiori for Web pattern. We choose the ListReport floorplan, and indicate that its data is loaded from an SDT.

Remember that the SAPMaterialGetList process returns the list of materials and the list of possible errors. The list of materials is handled internally to update our products, so we take the Messages SDT as the data source.

This web panel will show the collection of errors, if any, and process the list of materials.

We remove the actions on the grid, and check the corresponding properties to include it in the Fiori lanunchpad and in the master page menu.

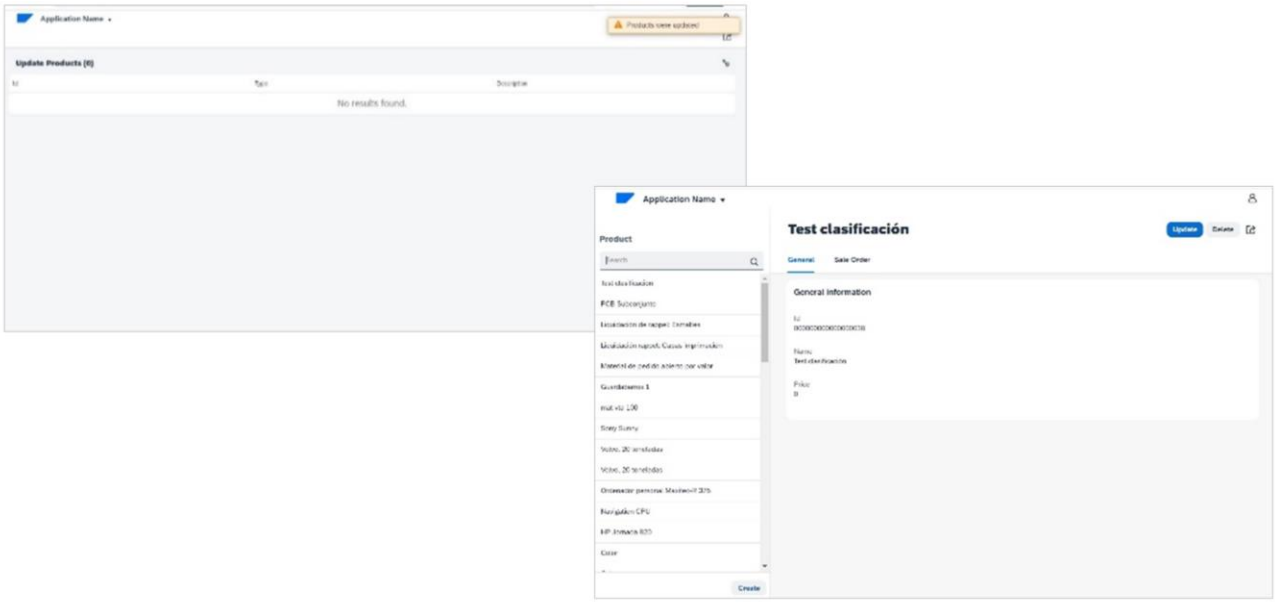
We save. Now let's go to its events.

When this object is opened, we want to call the process that invokes the GetList method, show the errors, if any, and process the list of materials.

Therefore, in the Start event we define the call to the SAPMaterialGetList procedure, defining the necessary variables.

Then we check if there were any errors and process the materials with messages indicating if there were any failures or if it was performed correctly.

Running the GetList method



Before running, we must delete the test data to load the products from the materials returned by the ERP. We press F5.

First, we check that no products have been loaded.

And now we go to the web panel to load the materials returned by the ERP.

We see that it was executed correctly and no errors or warnings were returned. Finally, we go back to the products to check that they were loaded correctly. Next, we will work with sales orders.

GeneXus by Globant

GeneXus™
by Globant

training.genexus.com