TrakSYS™ Operations Management Software

TrakSYS™ 8.0 Advanced Training Course Lab Manual

Revised 01-Sep-2013

Parsec Automation Corp.
180 North Riverview, No. 300
Anaheim, CA 92808 USA
Phone +1 714 996 5302
Fax +1 714 996 1845
www.parsec-corp.com

Contents

Items and Locations

Assignment

Create definitions for Locations, Items, and Item Log entries used to track the status and usage of various containers in the production environment. Utilize existing WEBTrak dashboards to create a historical record for a specific material container used at a Receiving Dock.

Instructions

1. Open **MODELTrak** and select the **PRODUCTTrak** panel.
2. In the **Locations** folder, add the following **New Locations** as child Locations under the existing **Receiving** Location:

Name	Unique Identifier
Dock 1	LOC.DOC.1
Dock 2	LOC.DOC.2

3. Add the following **New Locations** as child Locations under the existing **Inventory** Location:

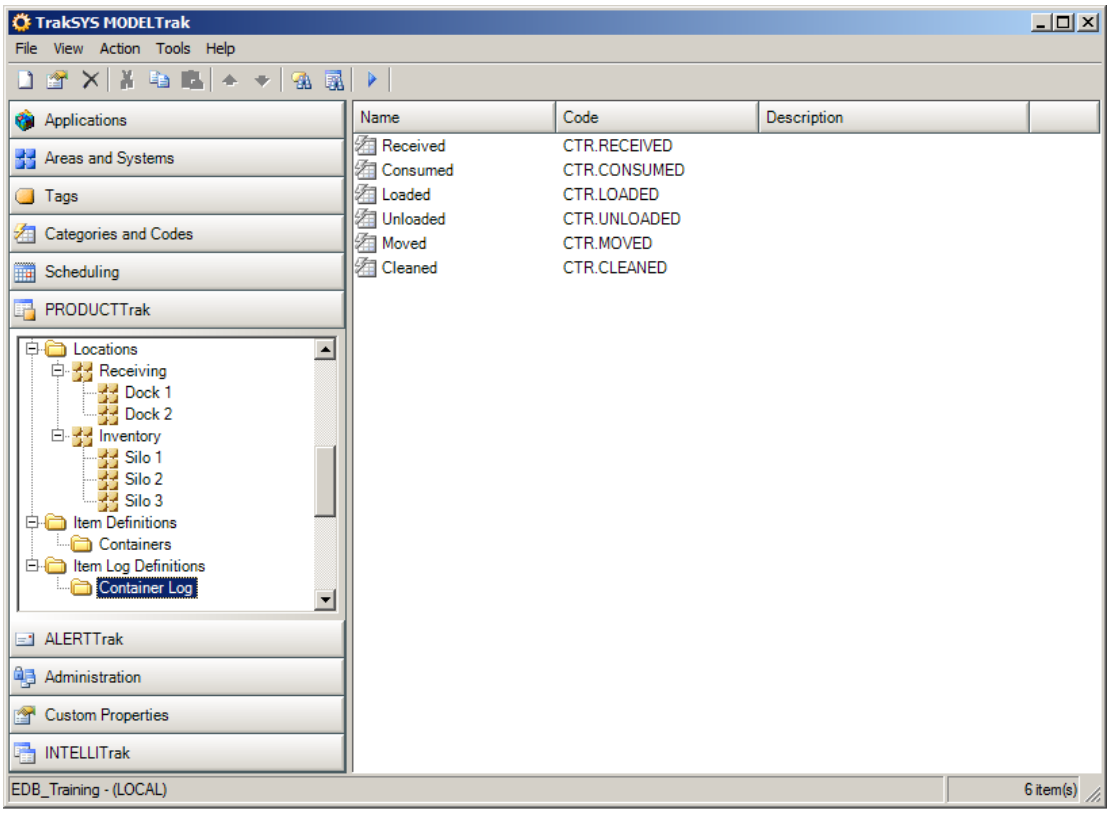
Name	Unique Identifier
Silo 1	LOC.SILO.1
Silo 2	LOC.SILO.2
Silo 3	LOC.SILO.3

4. In the **Item Definitions** folder, create a **New Item Definition Group** named **Containers**.
5. In the Containers group, add the following **New Item Definitions** to the group:

Name	Code	Quantity Units
Drum	CTR.DRUM	g
Tote	CTR.TOTE	g
Pallet	CTR.PALLET	g
Bag	CTR.BAG	g

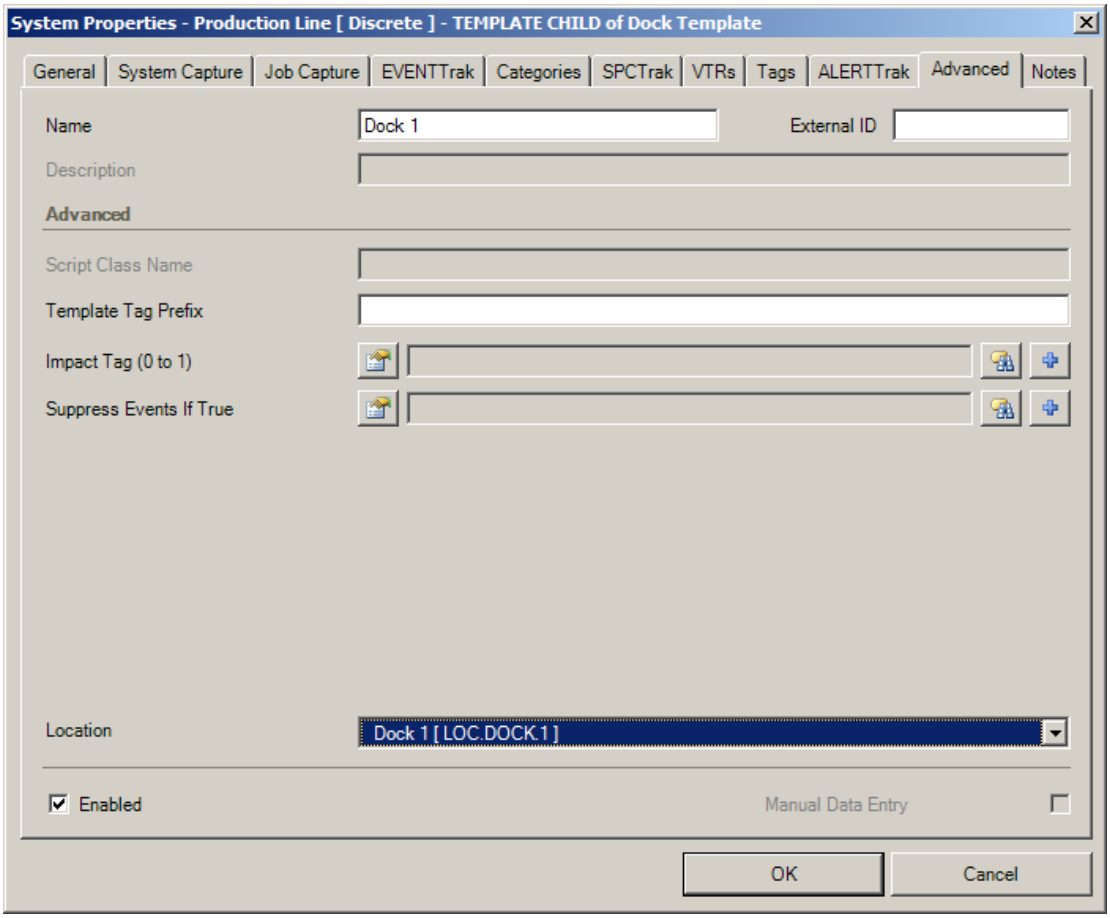
6. In the **Item Log Definitions** folder, create a **New Item Log Definition Group** named **Container Log**.
7. In the Container Log group, add the following **New Item Log Definitions** to the group:

Name	Code
Received	CTR.RECEIVED
Consumed	CTR.CONSUMED
Loaded	CTR.LOADED
Unloaded	CTR.UNLOADED
Moved	CTR.MOVED
Cleaned	CTR.CLEANED



8. Go to the **Areas and Systems** panel and assign the following **Locations** to the corresponding **Systems**. Location assignments are made on the **Advanced** tab of the **System Properties** dialog:

Area	System	Location
Receiving	Dock 1	Dock 1
Receiving	Dock 2	Dock 2



9. Open **WEBTrak** and login with the following account:

Login	administrator
Password	sa

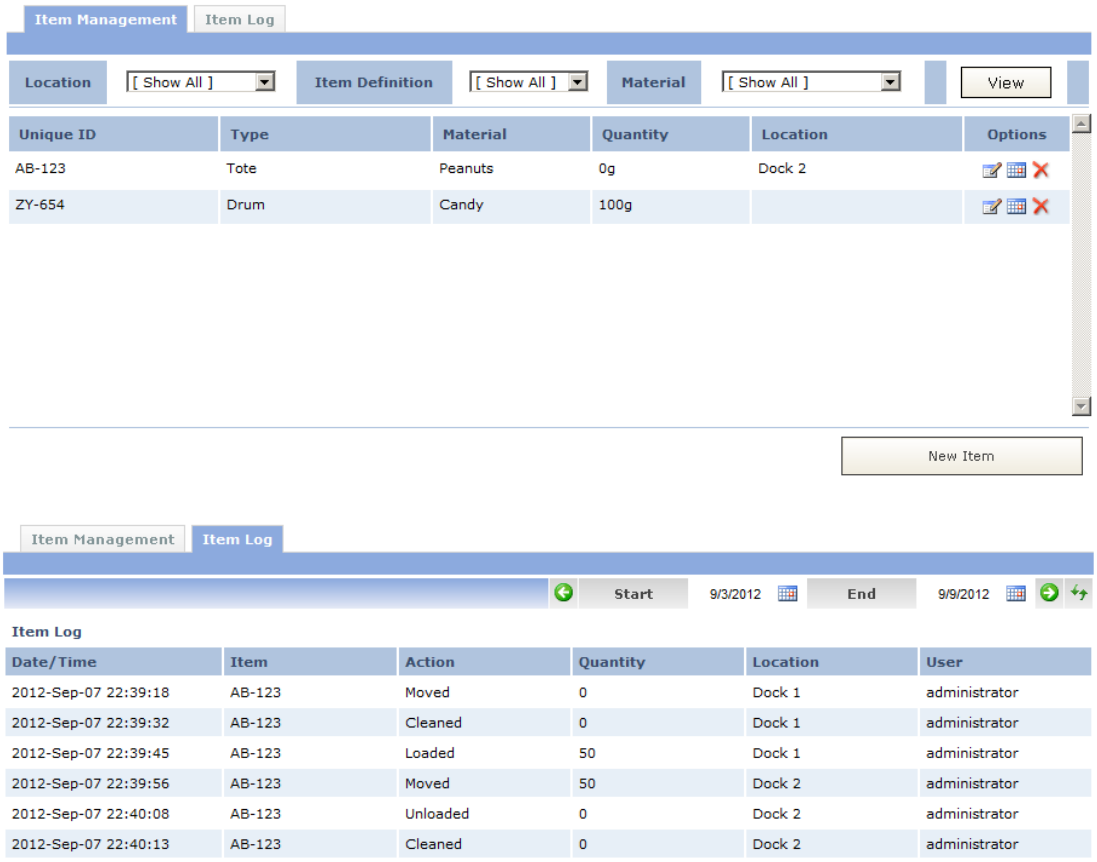
10. Navigate to the Report Group **TrakSYS Reports** | **TrakSYS Advanced Training** | **Lab: Items and Locations**.
11. Open the **Item Management** page and use the **New Item** button at the bottom of the page to add the following Items:

UniqueID	Item Definition	Material	Lot	Quantity	Location
AB-123	Tote	Peanuts	PZ-1053	0	[None]
ZY-654	Drum	Candy	CX-1837	100	[None]

12. In the list of Items, click the icon under **Options** for Tote **AB-123** to create the following entries in the **Item Log**:

Item Log Definition	Location	Quantity
Moved	Dock 1	0
Cleaned	Dock 1	0
Loaded	Dock 1	50
Moved	Dock 2	50
Unloaded	Dock 2	0
Cleaned	Dock 2	0

13. Click the Item Log tab at the top of the page to view a report of all entries recorded to the Item Log for a selected date range. The default date range is set to the current week.

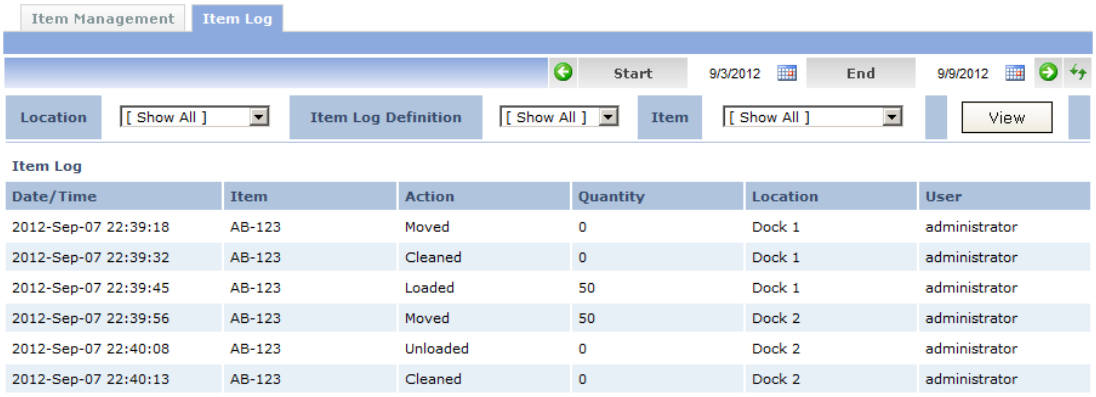


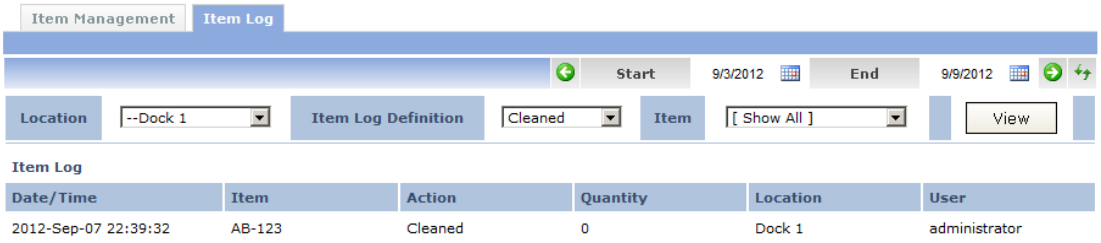
Advanced Assignment

Modify the Item Log report to include user filters for **Location**, **Item**, and **Item Log Definition**. User selections from these filters should update the results in the report page.

Tips and Hints

1. Review the HTML and queries of the **HTML Content** web part named **Filters** on the **Item Management** dashboard. This web part implements the filters for Location, Item Definition, and Material used in the Item List beneath it. This web part may be copied onto the Item Log dashboard and used as a starting point for the report filters. Important components of this web part include:
   a. The `<ets_dataset>` queries found on the **Data** tab when editing the web part's HTML.
   b. The `<ets_repeat>` and `{dataset[.@row]}` structures and Content Expressions found in the **HTML** for the web part.
   c. The `id` and `name` values associated with the `<select>` form elements in the web part's **HTML**.
2. Review the **Source SQL Query** for the **Item List** HTML Content web part on the **Item Management** dashboard. This example demonstrates how the Filter options are accessed in the query using **Content Expressions**. Important components of this SQL Query to review include:
   a. The SQL parameters defined at the top of the query (`@LocationID`, `@ItemDefinitionID`, `@MaterialID`).
   b. The Content Expressions assigned to these parameters in the **SET** statements at the top of the query.
   c. The use of the SQL parameters in the query's **WHERE** clause at the bottom of the query.
3. When editing web parts in a dashboard using the new Content Editor, use the **Preview** button to save any changes that have been made and view them immediately in a popup preview window.





Storage Systems

Assignment

Create a Storage System and several Transfer Definitions to track the movement of materials in and out of an Inventory location. The Storage System will be based on a System Template used to create a consistent model for all Silos used in the Inventory Area. Utilize existing WEBTrak dashboards to trigger transfers in from the different Receiving Docks and for emptying (setting) and cleaning the Silos.

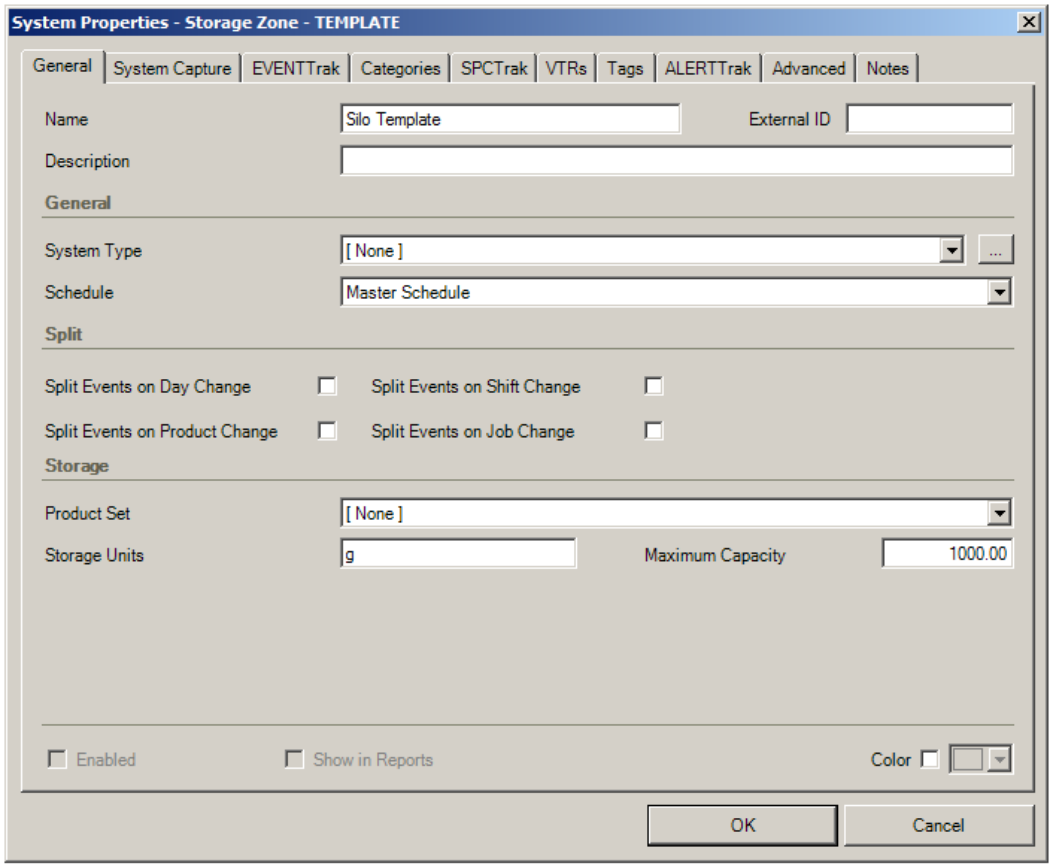
Instructions

1. Open **MODELTrak** and select the **Areas and Systems** panel.
2. Add a **New System Template [Storage]** named **Silo Template** to the **Inventory** Area with the following properties:
   a. **General**

Property	Value
Schedule	Master Schedule
Storage Units	g
Maximum Capacity	1000

   b. **Advanced**

Property	Value
Template Tag Prefix	ST.



3. Add the following **New Transfer Definition Groups** to the Silo Template. Groups are used to organize various types of functional definitions into common groupings:

Name	Description
Receiving	Transfers in from Receiving.
Staging	Transfers out to Staging.
Actions	Transfers set (applied) to the Silo.

4. Add the following **Transfer Definitions** to the **Receiving** Transfer Definition Group:
   a. Transfer Definition named **Receive from Dock 1**:
      i. **General**

Trigger Tag	ST.TRANSFER.IN.DOCK.1
Transfer Direction	In
Source System	Dock 1
Quantity Tag	ST.RUN.QUANTITY
Capture Quantity on	Transfer End

      ii. **Capture**

Material Tag	ST.RUN.MATERIAL.CODE
Lot Tag	ST.RUN.LOT

b. Transfer Definition named **Receive from Dock 2**:
   i. **General**

Trigger Tag	ST.TRANSFER.IN.DOCK.2
Transfer Direction	In
Source System	Dock 2
Quantity Tag	ST.RUN.QUANTITY
Capture Quantity on	Transfer End

   ii. **Capture**

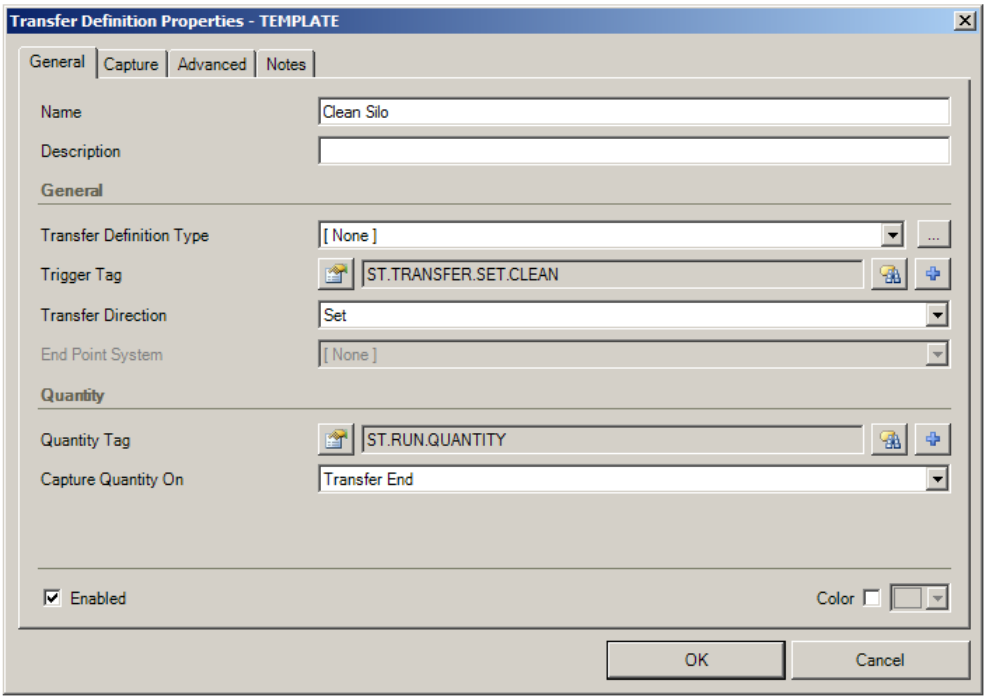
Material Tag	ST.RUN.MATERIAL.CODE
Lot Tag	ST.RUN.LOT

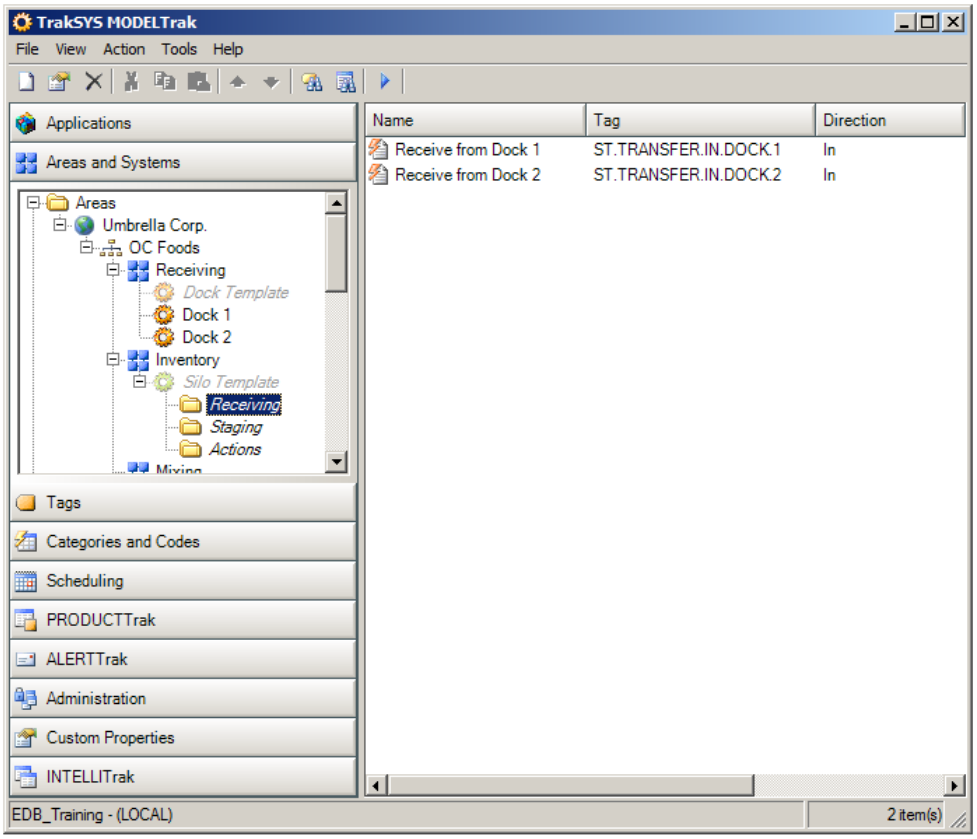
5. Add the following Transfer Definition to the **Actions** Transfer Definition Group:
   a. Transfer Definition named **Clean Silo**:
      i. **General**

Trigger Tag	ST.TRANSFER.SET.CLEAN
Transfer Direction	Set
Quantity Tag	ST.RUN.QUANTITY
Capture Quantity on	Transfer End

      ii. **Capture**

Material Tag	ST.RUN.MATERIAL.CODE
Lot Tag	ST.RUN.LOT





6. Right-click on the Silo Template and select the option **Create New System from Template**. This will create a new Child System based on the Template's structure named **New System (from Silo Template)**.

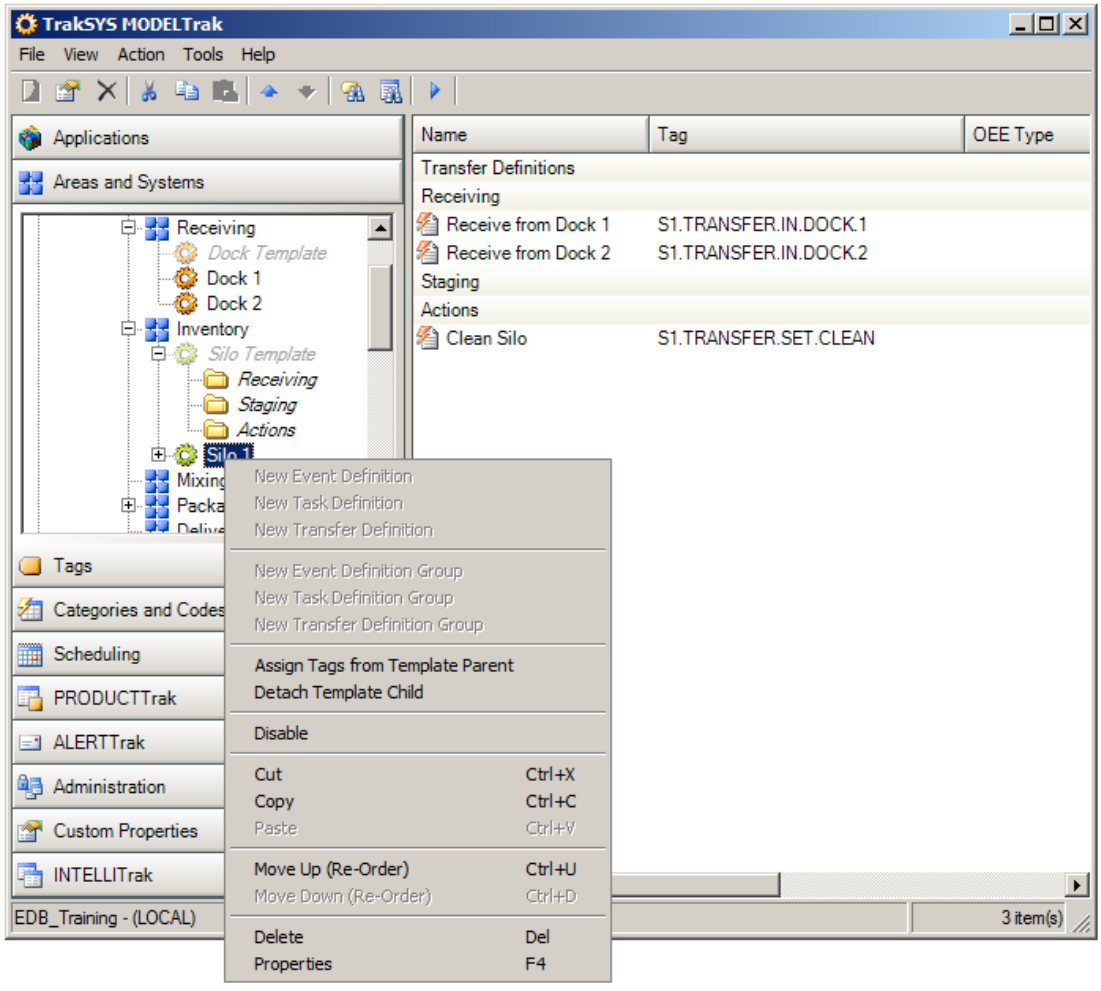
7. Modify the **New System (from Silo Template)** by changing the following properties:
   a. **General**

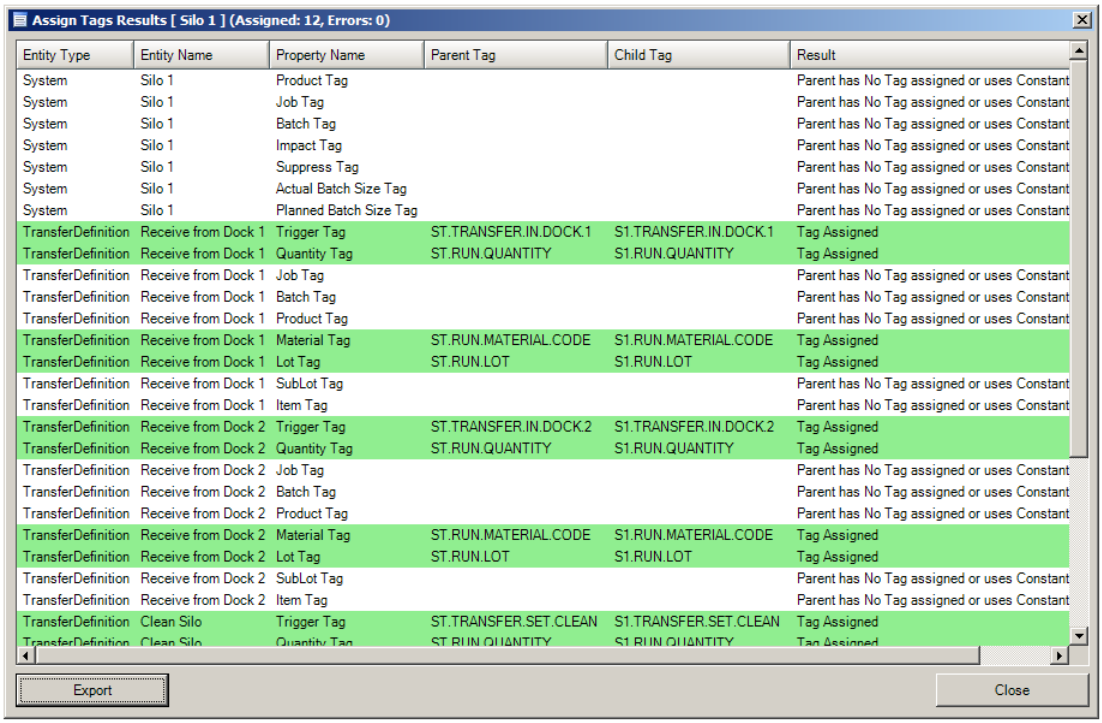
Property	Value
Name	Silo 1

   b. **Advanced**

Property	Value
Template Tag Prefix	S1.
Location	Silo 1 [LOC.SILO.1]

8. Right-click on the **Silo 1** System and select the option **Assign Tags from Template Parent**. This will automatically assign all required Tags to the System based on the Tag naming convention used for the Parent Template.





Entity Type	Entity Name	Property Name	Parent Tag	Child Tag	Result
System	Silo 1	Product Tag			Parent has No Tag assigned or uses Constant
System	Silo 1	Job Tag			Parent has No Tag assigned or uses Constant
System	Silo 1	Batch Tag			Parent has No Tag assigned or uses Constant
System	Silo 1	Impact Tag			Parent has No Tag assigned or uses Constant
System	Silo 1	Suppress Tag			Parent has No Tag assigned or uses Constant
System	Silo 1	Actual Batch Size Tag			Parent has No Tag assigned or uses Constant
System	Silo 1	Planned Batch Size Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 1	Trigger Tag	ST.TRANSFER.IN.DOCK.1	S1.TRANSFER.IN.DOCK.1	Tag Assigned
TransferDefinition	Receive from Dock 1	Quantity Tag	ST.RUN.QUANTITY	S1.RUN.QUANTITY	Tag Assigned
TransferDefinition	Receive from Dock 1	Job Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 1	Batch Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 1	Product Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 1	Material Tag	ST.RUN.MATERIAL.CODE	S1.RUN.MATERIAL.CODE	Tag Assigned
TransferDefinition	Receive from Dock 1	Lot Tag	ST.RUN.LOT	S1.RUN.LOT	Tag Assigned
TransferDefinition	Receive from Dock 1	SubLot Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 1	Item Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 2	Trigger Tag	ST.TRANSFER.IN.DOCK.2	S1.TRANSFER.IN.DOCK.2	Tag Assigned
TransferDefinition	Receive from Dock 2	Quantity Tag	ST.RUN.QUANTITY	S1.RUN.QUANTITY	Tag Assigned
TransferDefinition	Receive from Dock 2	Job Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 2	Batch Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 2	Product Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 2	Material Tag	ST.RUN.MATERIAL.CODE	S1.RUN.MATERIAL.CODE	Tag Assigned
TransferDefinition	Receive from Dock 2	Lot Tag	ST.RUN.LOT	S1.RUN.LOT	Tag Assigned
TransferDefinition	Receive from Dock 2	SubLot Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Receive from Dock 2	Item Tag			Parent has No Tag assigned or uses Constant
TransferDefinition	Clean Silo	Trigger Tag	ST.TRANSFER.SET.CLEAN	S1.TRANSFER.SET.CLEAN	Tag Assigned
TransferDefinition	Clean Silo	Quantity Tag	ST.RUN.QUANTITY	S1.RUN.QUANTITY	Tag Assigned

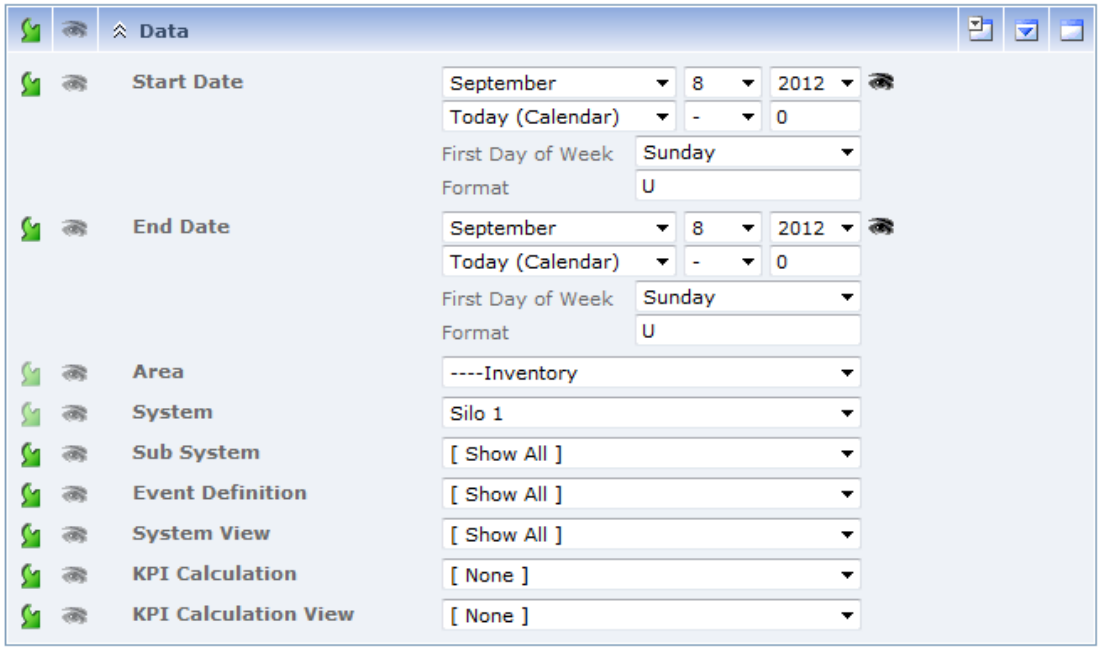
9. Restart the **LOGICTrak Service** for the configuration changes to take effect.
10. Open **WEBTrak** and login with the following account:

Login	administrator
Password	sa

11. Navigate to the Report Group **TrakSYS Reports** | **TrakSYS Advanced Training** | **Lab: Storage Systems**.
12. Open the **Design** view (Report Design page) for the **Silo Overview** dashboard and update the following report parameters in the **Data** section:

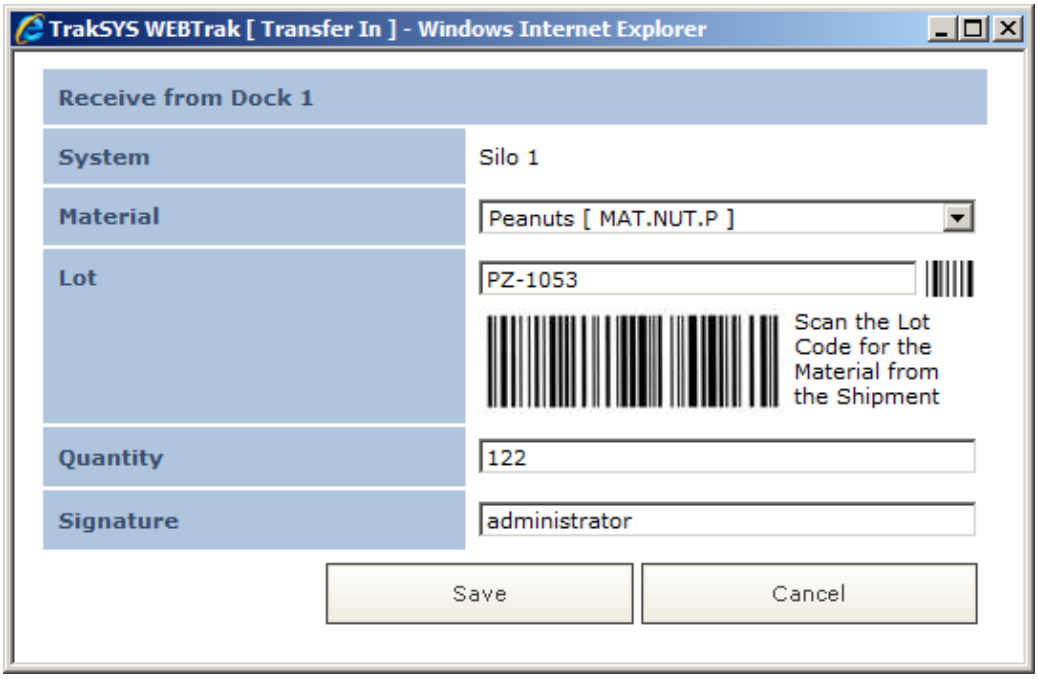
Parameter	Value
Area	Inventory
System	Silo 1

a. Be sure to click the **Save** button to commit changes made in the Report Design page.



13. Open the **Silo Overview** dashboard. Click the buttons on the left side of the dashboard to initiate the Transfers configured for the Storage System. Enter the following information for each Transfer initiated:
    a. **Receive from Dock 1**

Material	Peanuts [MAT.NUT.P]
Lot	PZ-1053
Quantity	122



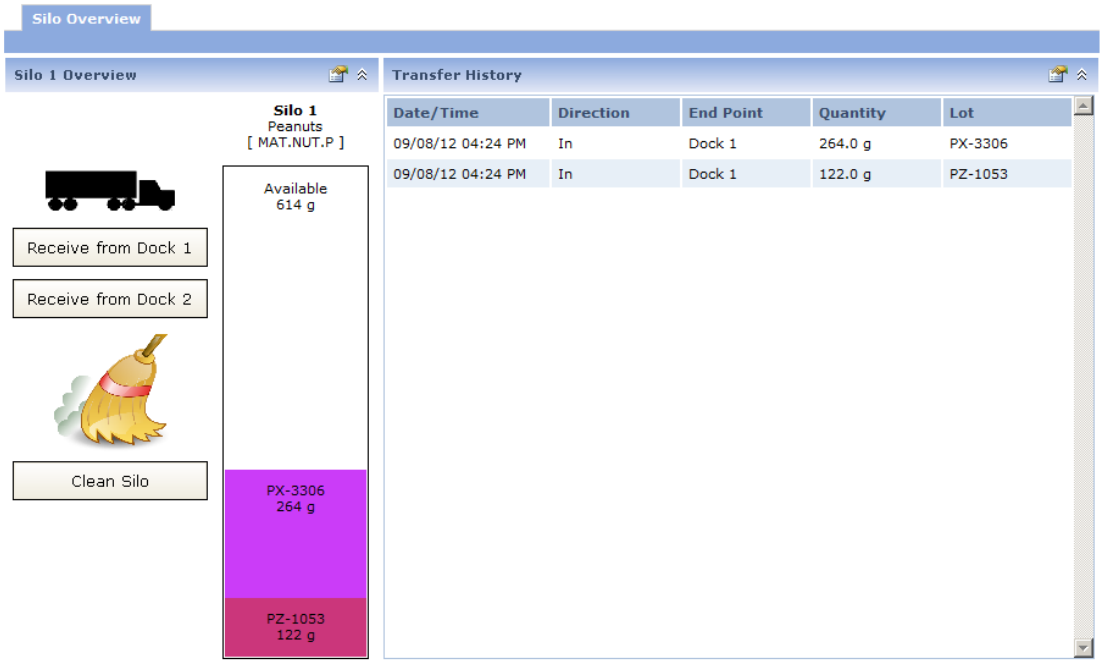
b. **Receive from Dock 2**

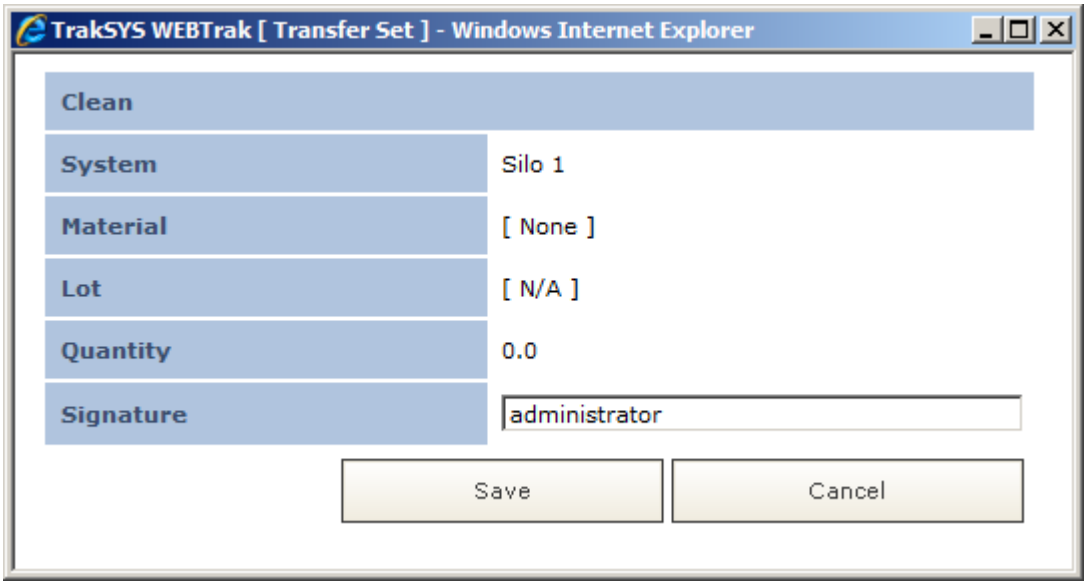
Material	Peanuts [MAT.NUT.P]
Lot	PX-3306
Quantity	264

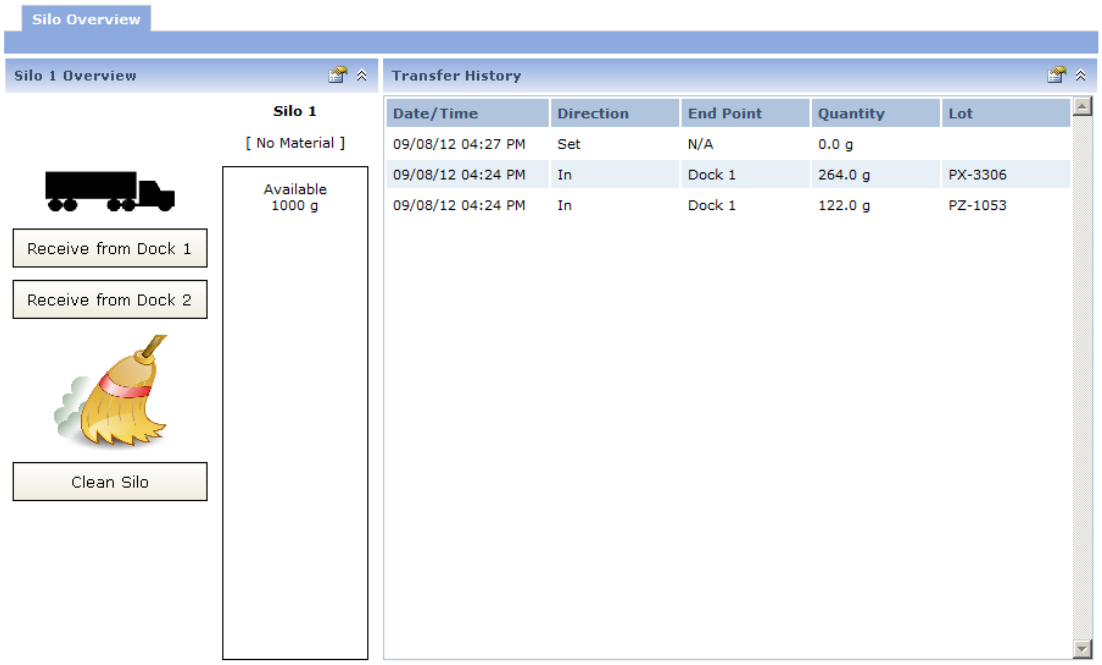
c. **Clean Silo**

i. No data entry required

14. Confirm that entries are made in the **Transfer History** list after each Transfer is performed.





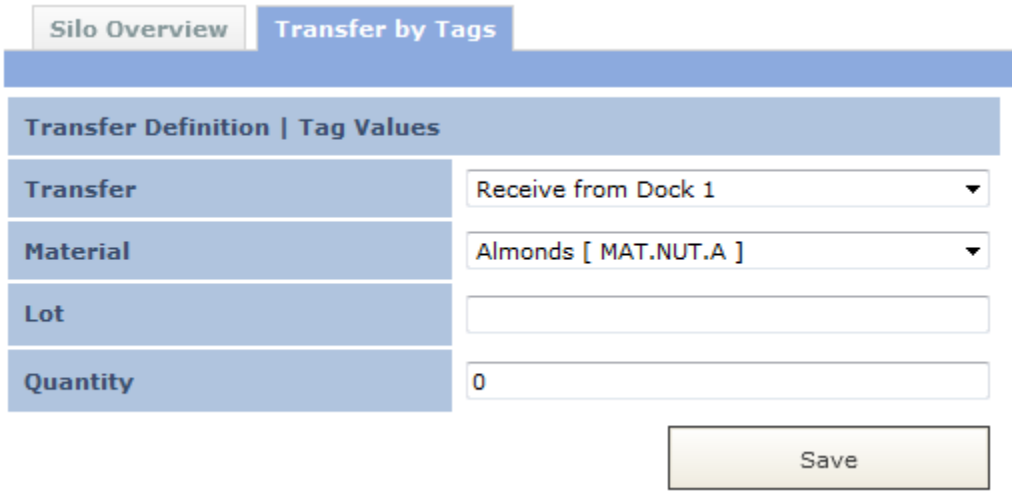


Advanced Assignment

Create an **Item Log** entry when performing a Transfer between an Item and a Storage System. Update the configuration for the Receiving Transfer Definitions by assigning an **Item Tag** and **Item Log Definition** to each Transfer. Execute the Transfer by updating the corresponding Tag values from a WEBTrak dashboard. Check the Item Log results with the Item Log dashboard used in the Items and Locations lab.

Tips and Hints

1. Utilize the existing Tag named **S1.RUN.ITEM.UNIQUEID** as the Item Tag assigned to the Transfer Definitions.
2. Update the Area and System parameters in the Report Design page for the dashboard **Transfer by Tags** in the Report Group **Lab: Storage Systems**. The System should be set to **Silo 1** for the dashboard to function properly.
3. Finish customizing the web part name **Transfer Definition Tags** on the dashboard **Transfer by Tags** to update the necessary Tags for triggering a Transfer.
   a. Add an additional textbox to the form named **itemUniqueID** for entering the Unique ID of the Item to use in the transfer
      i. Optionally, create a `<select>` list instead of a textbox that is populated with the UniqueIDs of all existing Items using a `<ets_dataset>` query.
   b. Finish the query for the **Post Back SQL** by adding the following:
      i. Query to retrieve all associated Tags for the selected Transfer Definition
      ii. `UPDATE` statements for table **tTag** to update all necessary Tag values with data entered in the web form. The fields to update are `tTag.Value` and `tTag.UpdateDateTime`.
4. Restart the **LOGICTrak Service** so that Tag value changes will be processed.



Batch Systems and Recipes

Assignment

Create a Batch System, Sub-Systems, and Function Definitions to model the equipment units and capabilities of a Mixing Process Cell. The Batch System will be based on a System Template used to create a consistent model for all Mixing Process Cells in the Mixing Area. Add a Recipe to the Mixing Product Set in PRODUCTTrak to define the size, planned duration, material requirements, parameter values, and execution steps for creating a specific Product on the Mixing Process Cell.

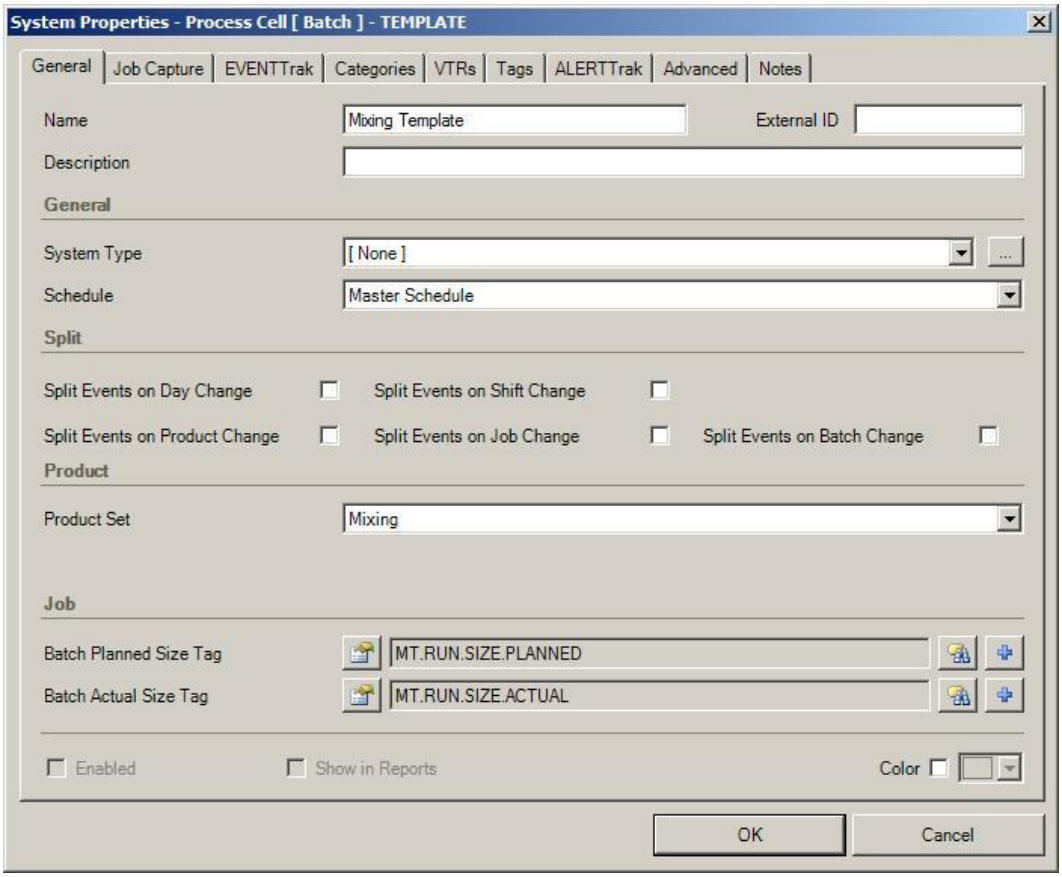
Instructions

1. Open **MODELTrak** and select the **Areas and Systems** panel.
2. Add a **New System Template [Batch]** named **Mixing Template** to the **Mixing** Area with the following properties:
   a. **General**

Property	Value
Schedule	Master Schedule
Product Set	Mixing
Batch Planned Size Tag	MT.RUN.SIZE.PLANNED
Batch Actual Size Tag	MT.RUN.SIZE.ACTUAL

   b. **Advanced**

Property	Value
Template Tag Prefix	MT.

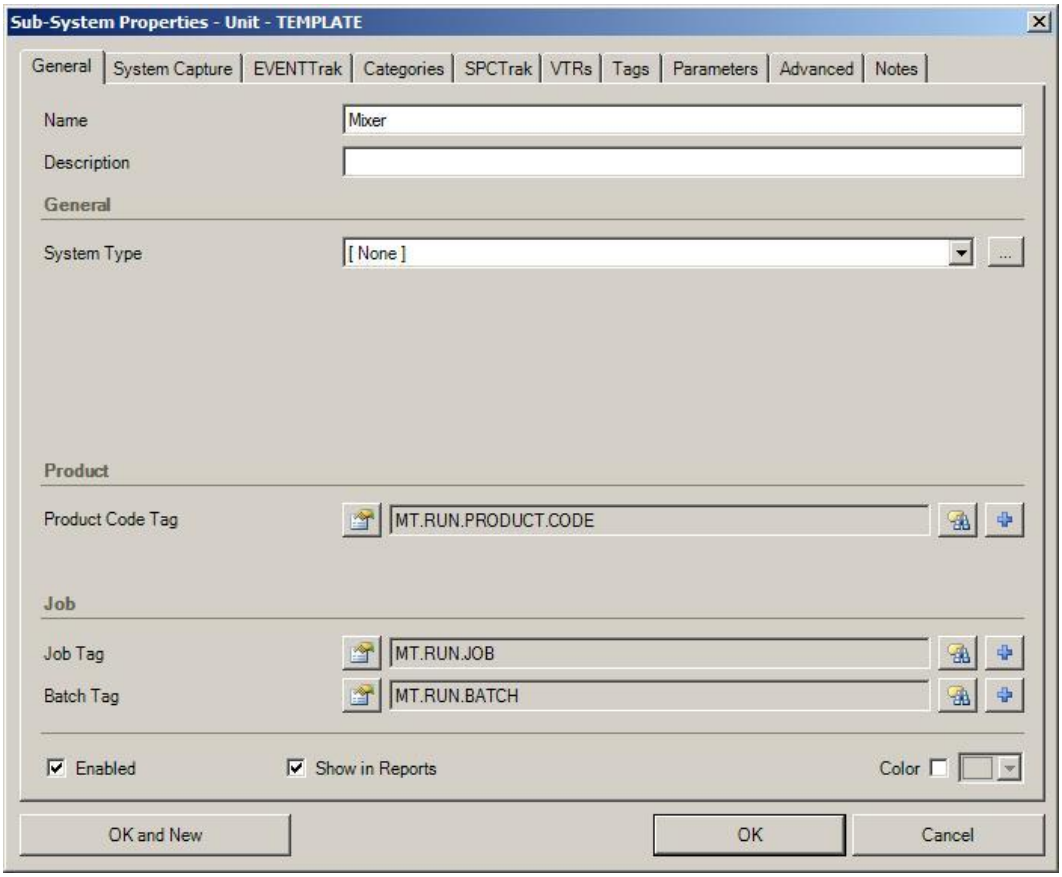


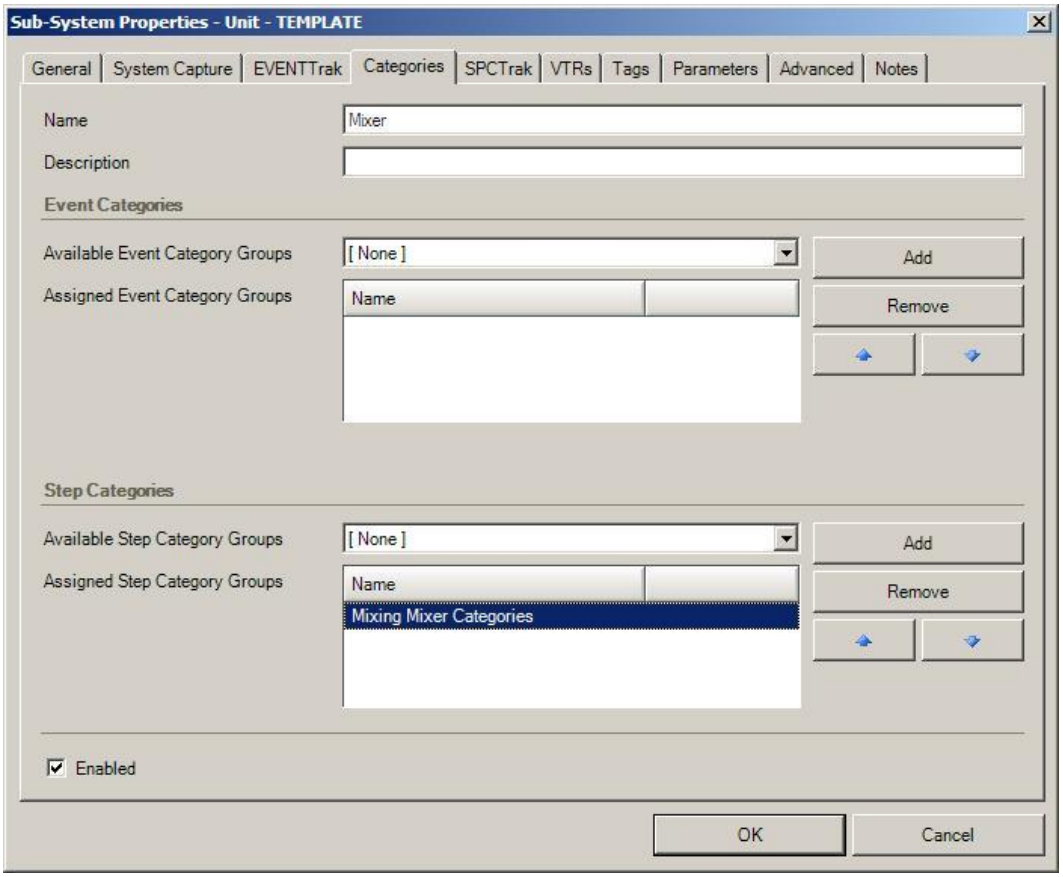
3. Add a **New Sub-System [Unit]** named **Mixer** to the **Mixing Template** System Template with the following properties:
   a. **General**

Property	Value
Product Code Tag	MT.RUN.PRODUCT.CODE
Job Tag	MT.RUN.JOB
Batch Tag	MT.RUN.BATCH

   b. **Categories**

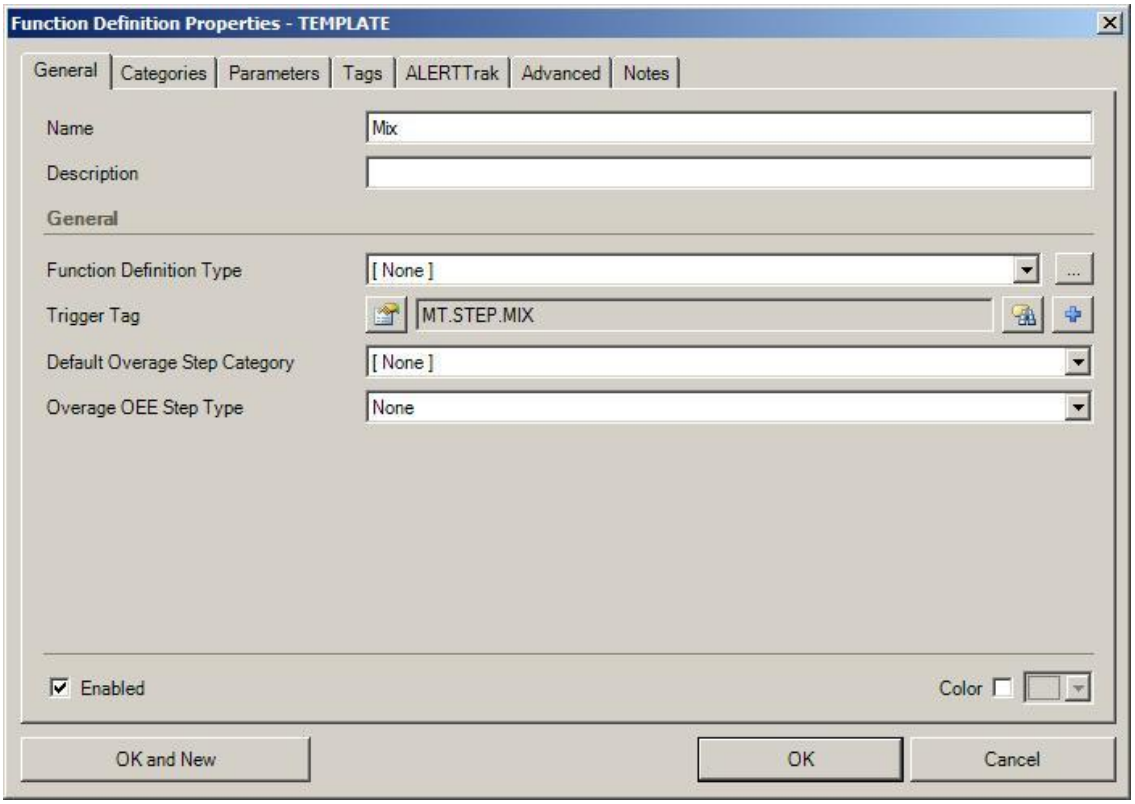
Property	Value
Assigned Step Category Groups	Mixing Mixer Categories

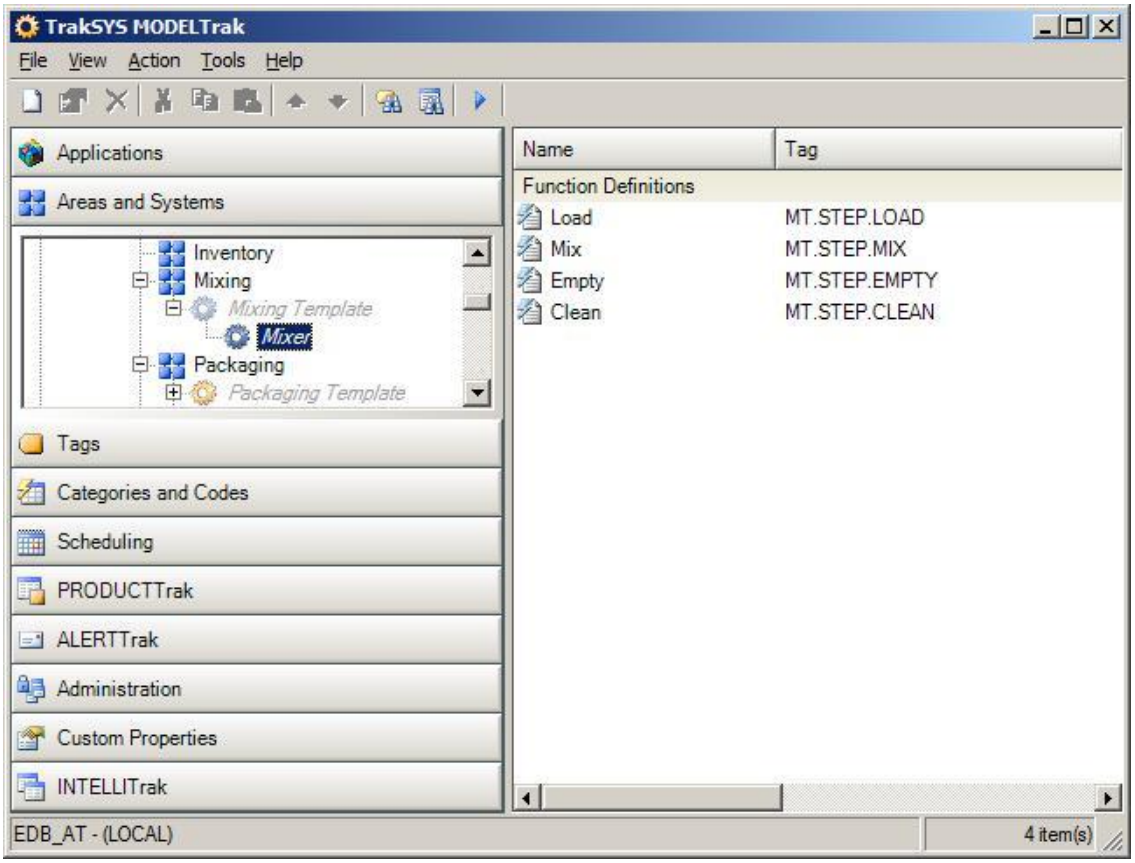

Sub-System Properties - Unit - TEMPLATE
General
System Capture
EVENTTrak
Categories
SPCTrak
VTRs
Tags
Parameters
Advanced
Notes
Name
Mixer
Description
General
System Type
[None]
Product
Product Code Tag
MT.RUN.PRODUCT.CODE
Job
Job Tag
MT.RUN.JOB
Batch Tag
MT.RUN.BATCH
Enabled
Show in Reports
Color
OK and New
OK
Cancel



4. Add the following **Function Definitions** to the **Mixer** Sub-System:

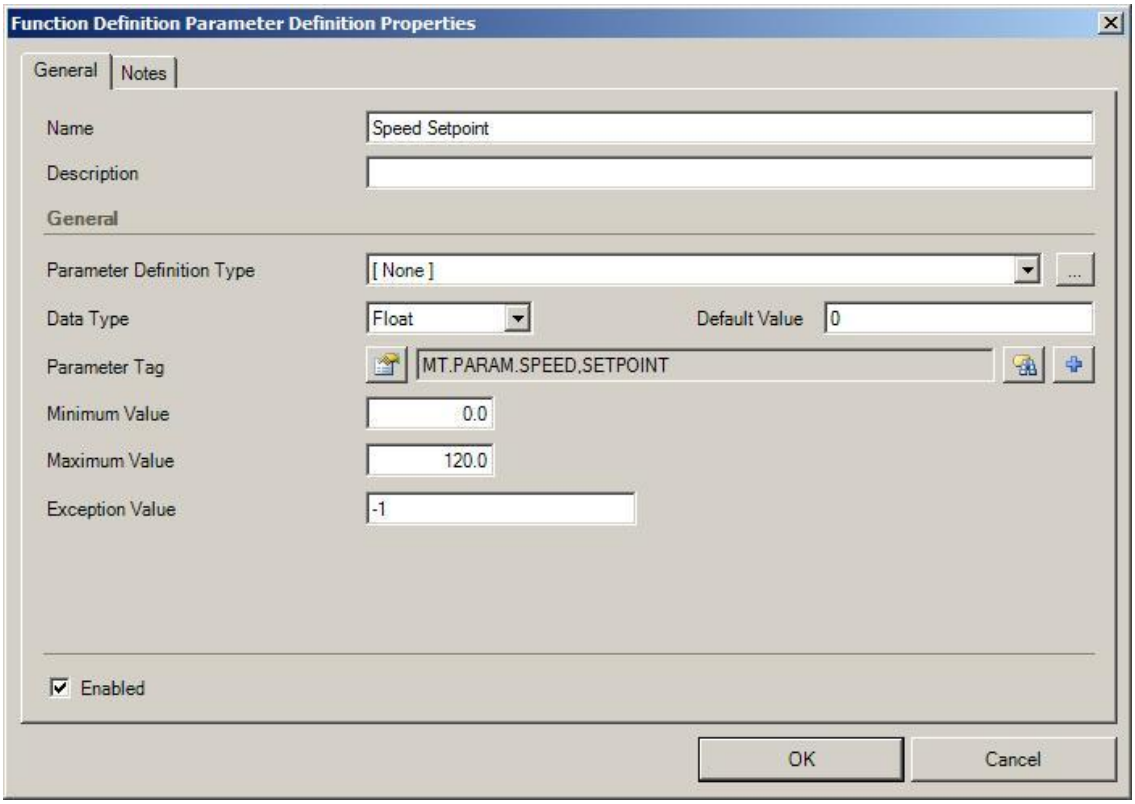
Name	Trigger Tag
Load	MT.STEP.LOAD
Mix	MT.STEP.MIX
Empty	MT.STEP.EMPTY
Clean	MT.STEP.CLEAN

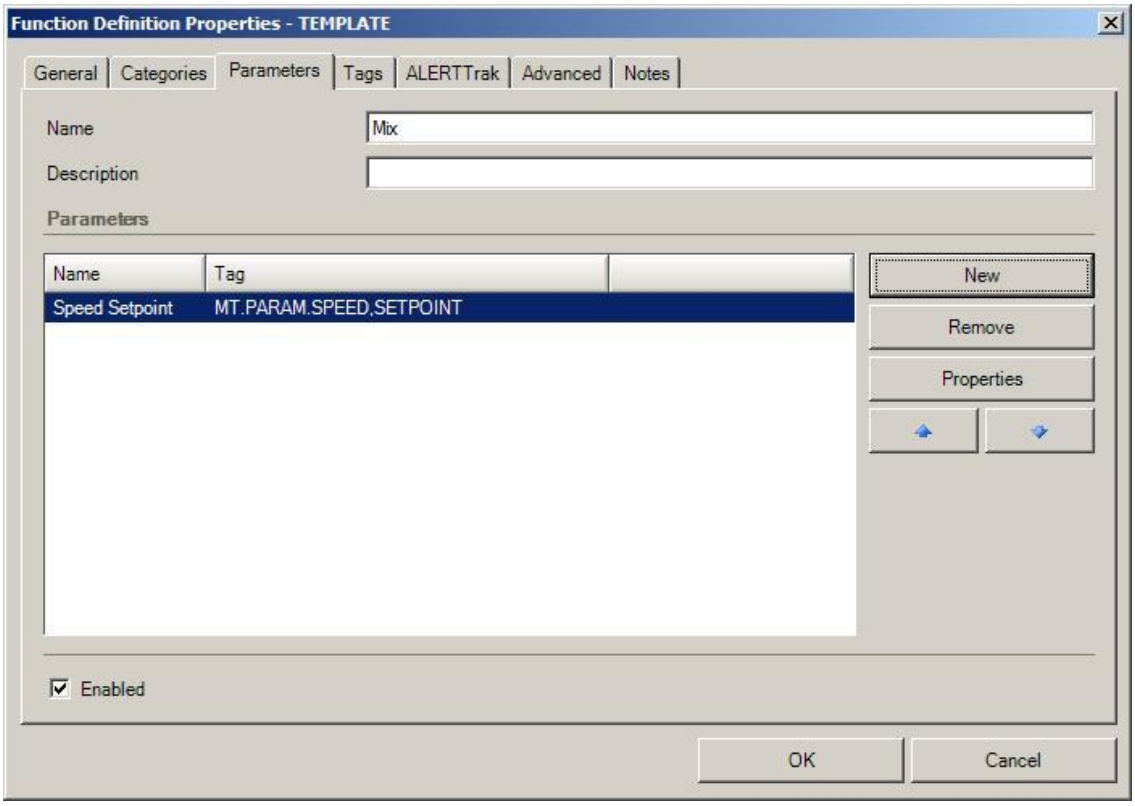




5. Add a **Parameter** named **Speed Setpoint** to the **Mix** Function Definition:

Data Type	Float
Parameter Definition Type	[None]
Default Value	0
Parameter Tag	MT.PARAM.SPEED.SETPOINT
Minimum Value	0.0
Maximum Value	120.0
Exception Value	-1





6. Add a **Parameter** named **Speed Setpoint** to the **Empty** Function Definition:

Data Type	Float
Parameter Definition Type	[None]
Default Value	0
Parameter Tag	MT.PARAM.SPEED.SETPOINT
Minimum Value	0.0
Maximum Value	120.0
Exception Value	-1

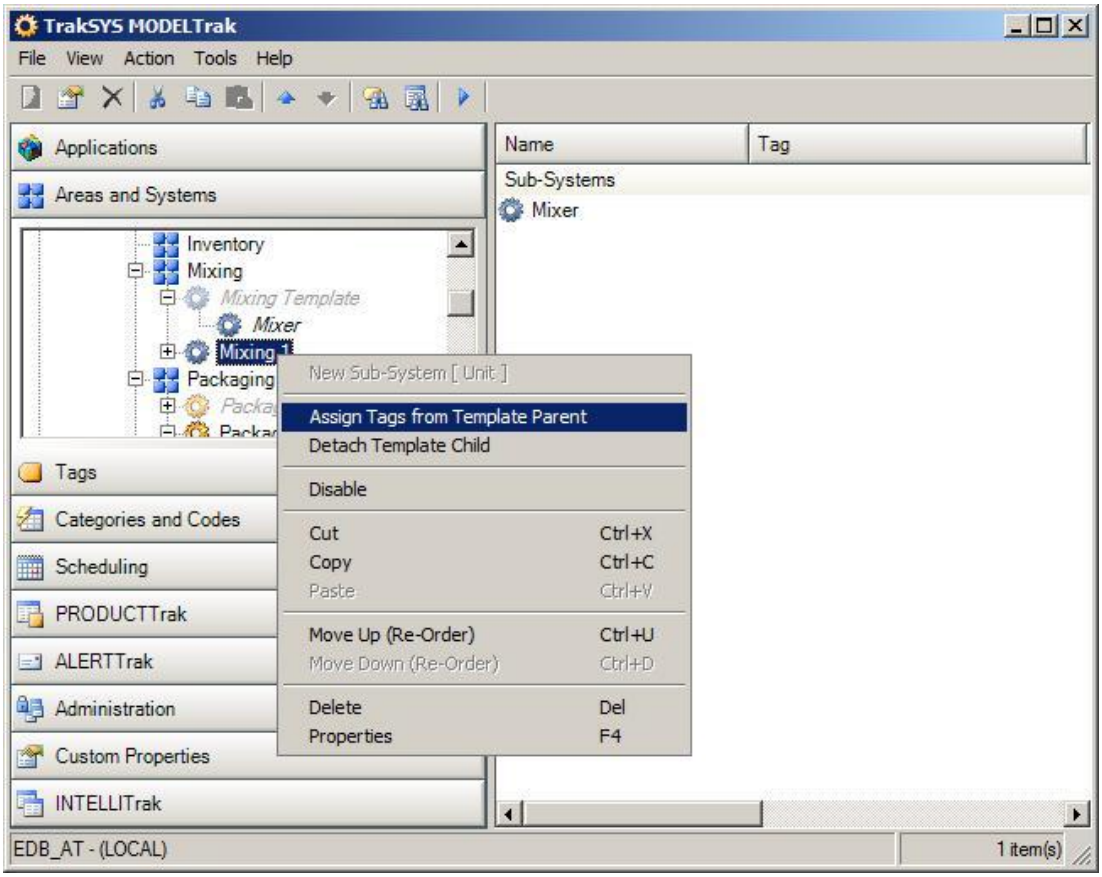
7. Right-click on the Mixing Template and select the option **Create New System from Template**. This will create a new Child System based on the Template's structure named **New System (from Mixing Template)**.
8. Modify the **New System (from Mixing Template)** by changing the following properties:
   a. **General**

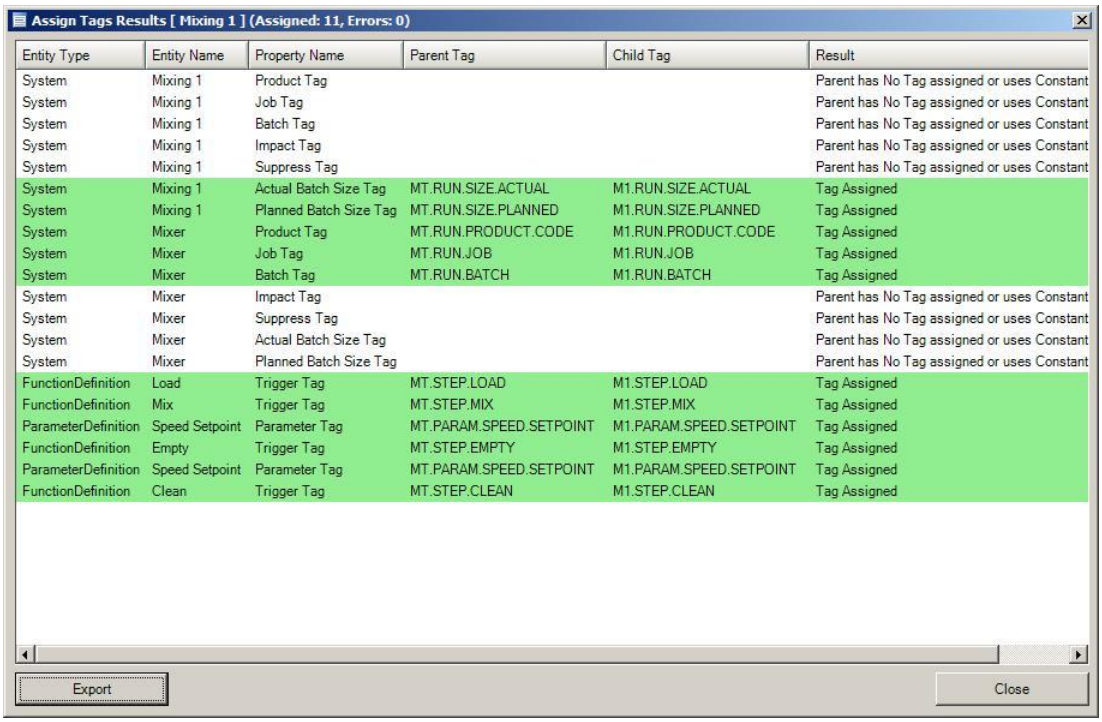
Property	**Value**
Name	Mixing 1

   b. **Advanced**

Property	**Value**
Template Tag Prefix	M1.

9. Right-click on the **Mixing 1** System and select the option **Assign Tags from Template Parent**. This will automatically assign all required Tags to the System based on the Tag naming convention used for the Parent Template.

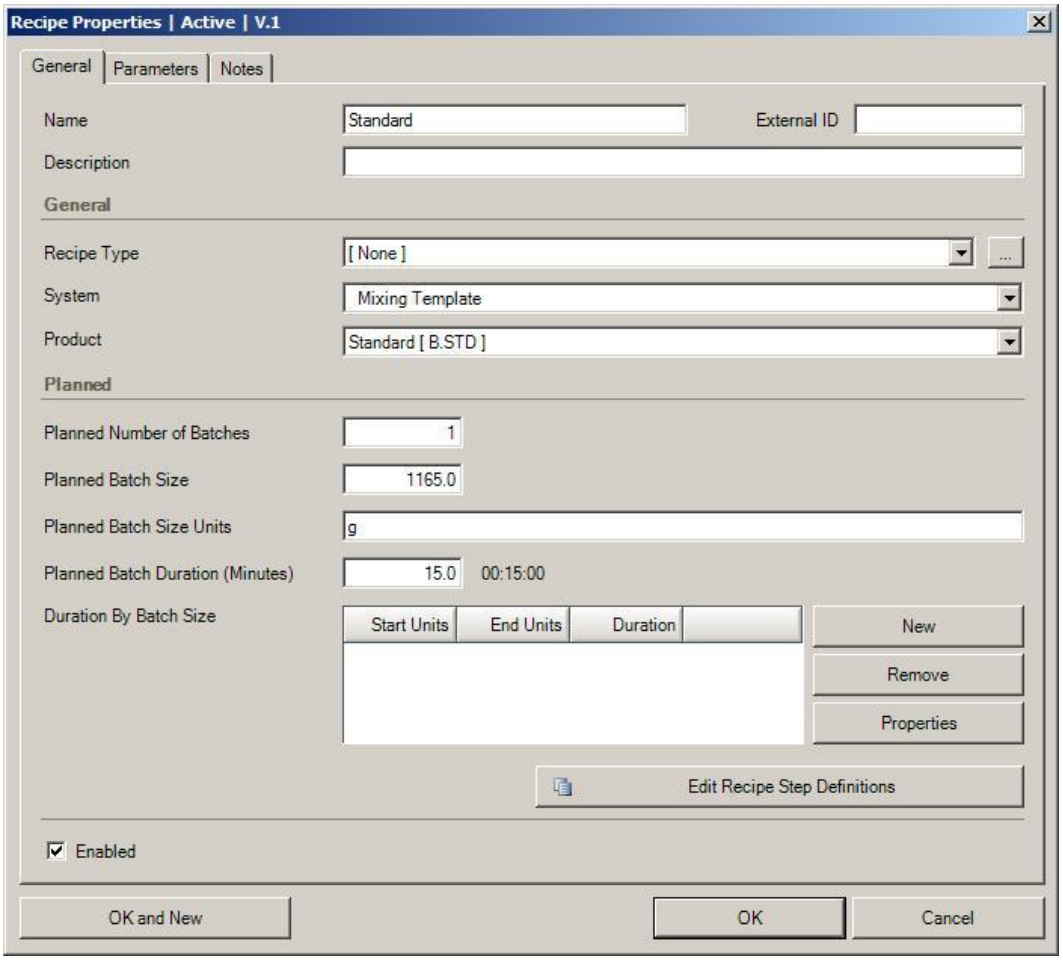




Entity Type	Entity Name	Property Name	Parent Tag	Child Tag	Result
System	Mixing 1	Product Tag			Parent has No Tag assigned or uses Constant
System	Mixing 1	Job Tag			Parent has No Tag assigned or uses Constant
System	Mixing 1	Batch Tag			Parent has No Tag assigned or uses Constant
System	Mixing 1	Impact Tag			Parent has No Tag assigned or uses Constant
System	Mixing 1	Suppress Tag			Parent has No Tag assigned or uses Constant
System	Mixing 1	Actual Batch Size Tag	MT.RUN.SIZE.ACTUAL	M1.RUN.SIZE.ACTUAL	Tag Assigned
System	Mixing 1	Planned Batch Size Tag	MT.RUN.SIZE.PLANNED	M1.RUN.SIZE.PLANNED	Tag Assigned
System	Mixer	Product Tag	MT.RUN.PRODUCT.CODE	M1.RUN.PRODUCT.CODE	Tag Assigned
System	Mixer	Job Tag	MT.RUN.JOB	M1.RUN.JOB	Tag Assigned
System	Mixer	Batch Tag	MT.RUN.BATCH	M1.RUN.BATCH	Tag Assigned
System	Mixer	Impact Tag			Parent has No Tag assigned or uses Constant
System	Mixer	Suppress Tag			Parent has No Tag assigned or uses Constant
System	Mixer	Actual Batch Size Tag			Parent has No Tag assigned or uses Constant
System	Mixer	Planned Batch Size Tag			Parent has No Tag assigned or uses Constant
FunctionDefinition	Load	Trigger Tag	MT.STEP.LOAD	M1.STEP.LOAD	Tag Assigned
FunctionDefinition	Mix	Trigger Tag	MT.STEP.MIX	M1.STEP.MIX	Tag Assigned
ParameterDefinition	Speed Setpoint	Parameter Tag	MT.PARAM.SPEED.SETPOINT	M1.PARAM.SPEED.SETPOINT	Tag Assigned
FunctionDefinition	Empty	Trigger Tag	MT.STEP.EMPTY	M1.STEP.EMPTY	Tag Assigned
ParameterDefinition	Speed Setpoint	Parameter Tag	MT.PARAM.SPEED.SETPOINT	M1.PARAM.SPEED.SETPOINT	Tag Assigned
FunctionDefinition	Clean	Trigger Tag	MT.STEP.CLEAN	M1.STEP.CLEAN	Tag Assigned

10. Navigate to the **PRODUCTTrak** section in MODELTrak.
11. Add a **New Recipe** named **Standard** to the **Mixing** Product Set with the following properties:
    a. **General**

Property	**Value**
Recipe Type	[None]
System	Mixing Template
Product	Standard [B.STD]
Planned Number of Batches	1
Planned Batch Size	1165.0
Planned Batch Size Units	g
Planned Batch Duration (Minutes)	15.0

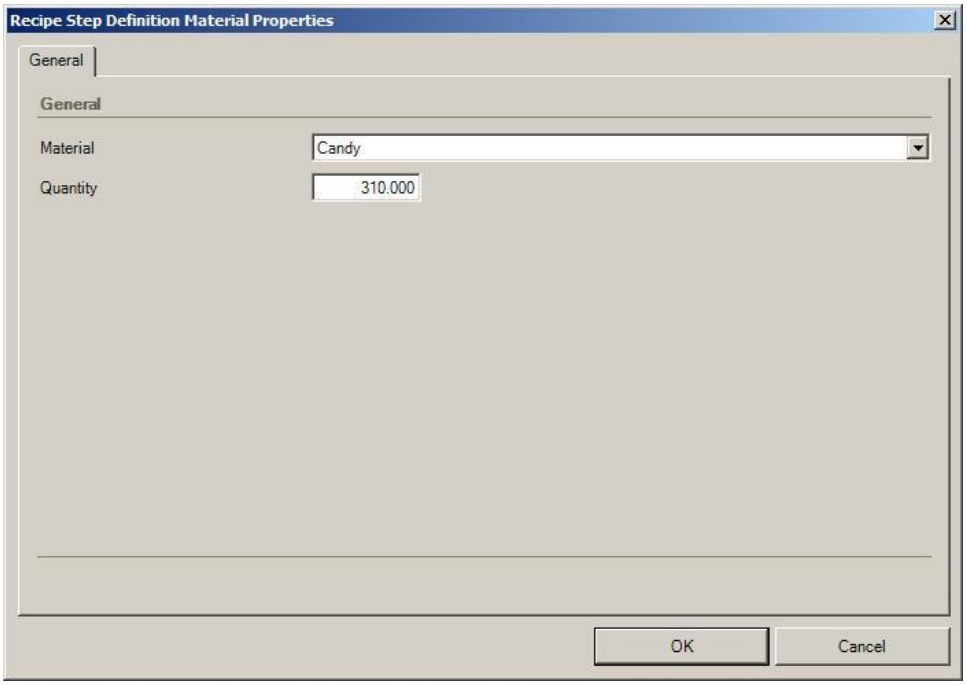


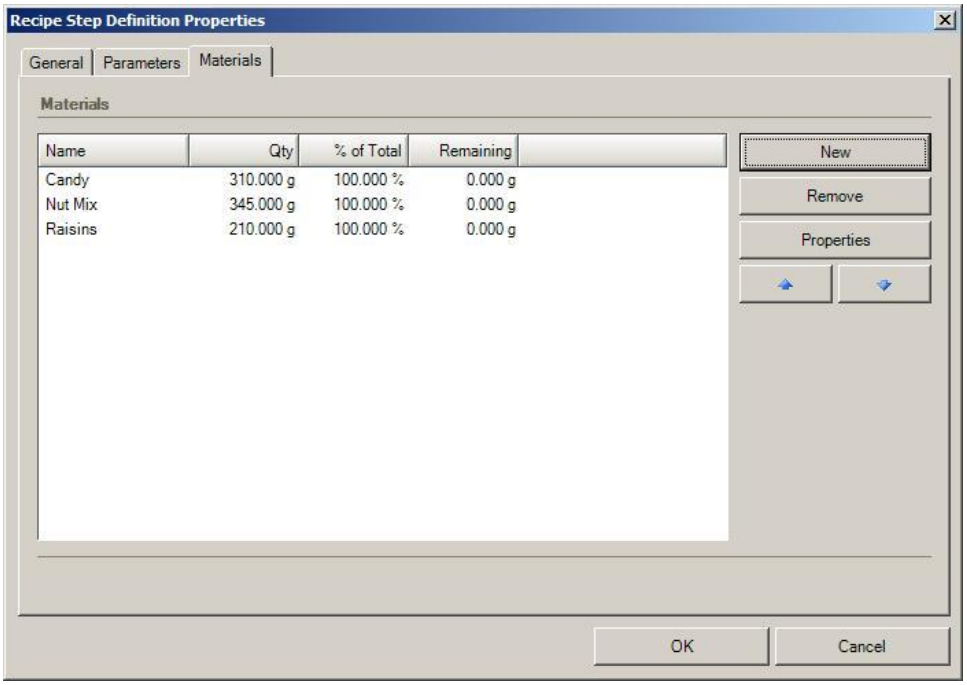
12. Click the **Edit Recipe Step Definitions** button for the **Standard** recipe.
13. Add several **New** Recipe Step Definitions with the following properties:
    a. Recipe Step Definition #1
        i. **General**

Function Definition	Load
Start Sequence	1
End Sequence	1
Planned Duration Seconds	120

        ii. **Materials**
            - Add 3 **New** Materials with the following properties:

Material	**Quantity**
Candy	310.000
Nut Mix	345.000
Raisins	210.000





b. Recipe Step Definition #2

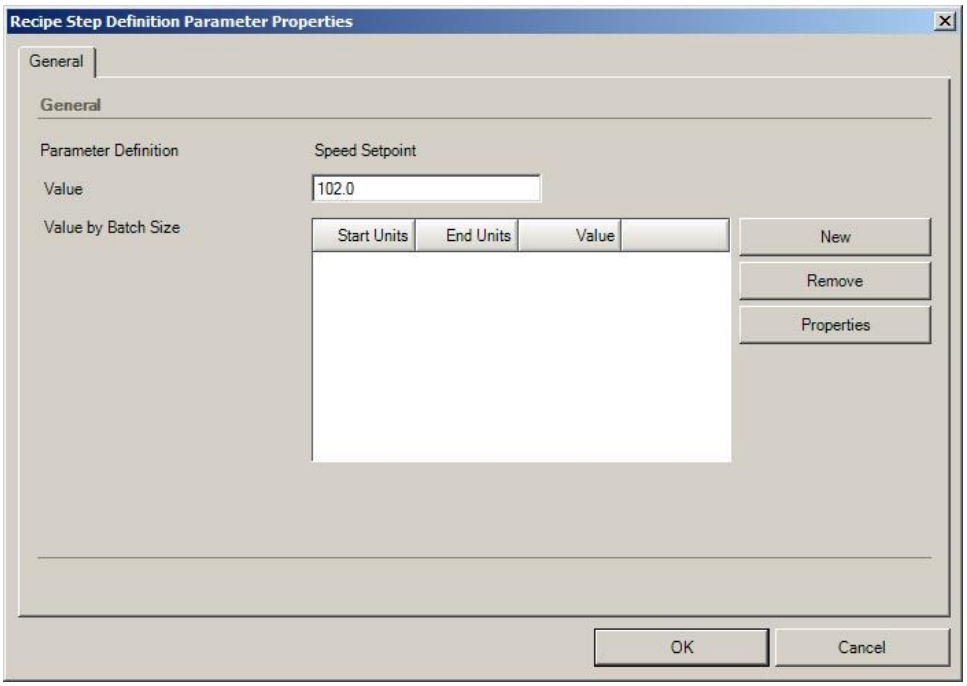
i. **General**

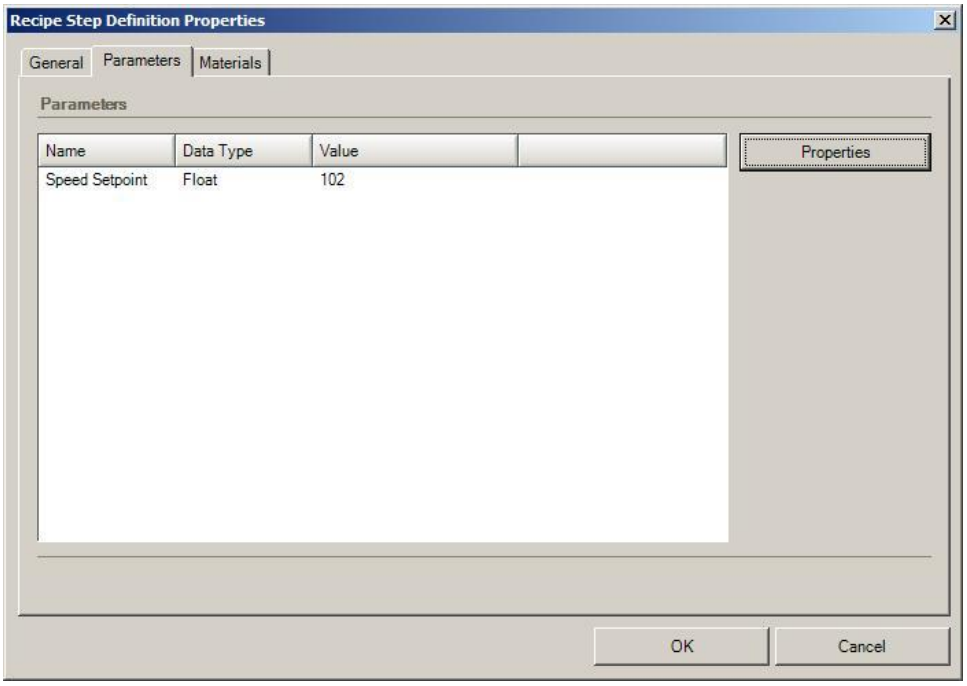
Function Definition	Mix
Start Sequence	2
End Sequence	2
Planned Duration Seconds	480

ii. **Parameters**

- Set the Properties for the following Parameters:

Parameter Definition	Value
Speed Setpoint	102





c. Recipe Step Definition #3

i. **General**

Function Definition	Empty
Start Sequence	3
End Sequence	3
Planned Duration Seconds	120

ii. **Parameters**

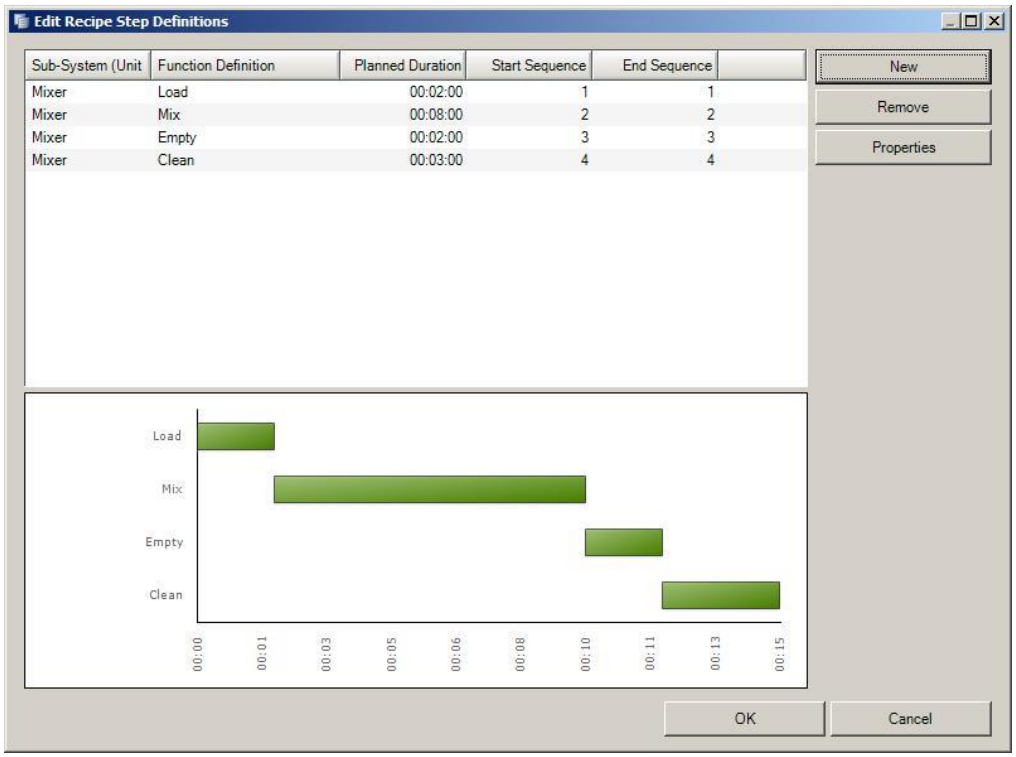
- Set the Properties for the following Parameters:

Parameter Definition	Value
Speed Setpoint	0

a. Recipe Step Definition #4

i. **General**

Function Definition	Clean
Start Sequence	4
End Sequence	4
Planned Duration Seconds	180



14. Restart the **LOGICTrak Service** for the configuration changes to take effect.

Advanced Assignment

There is no Advanced Assignment for this lab.

Batch Interfaces

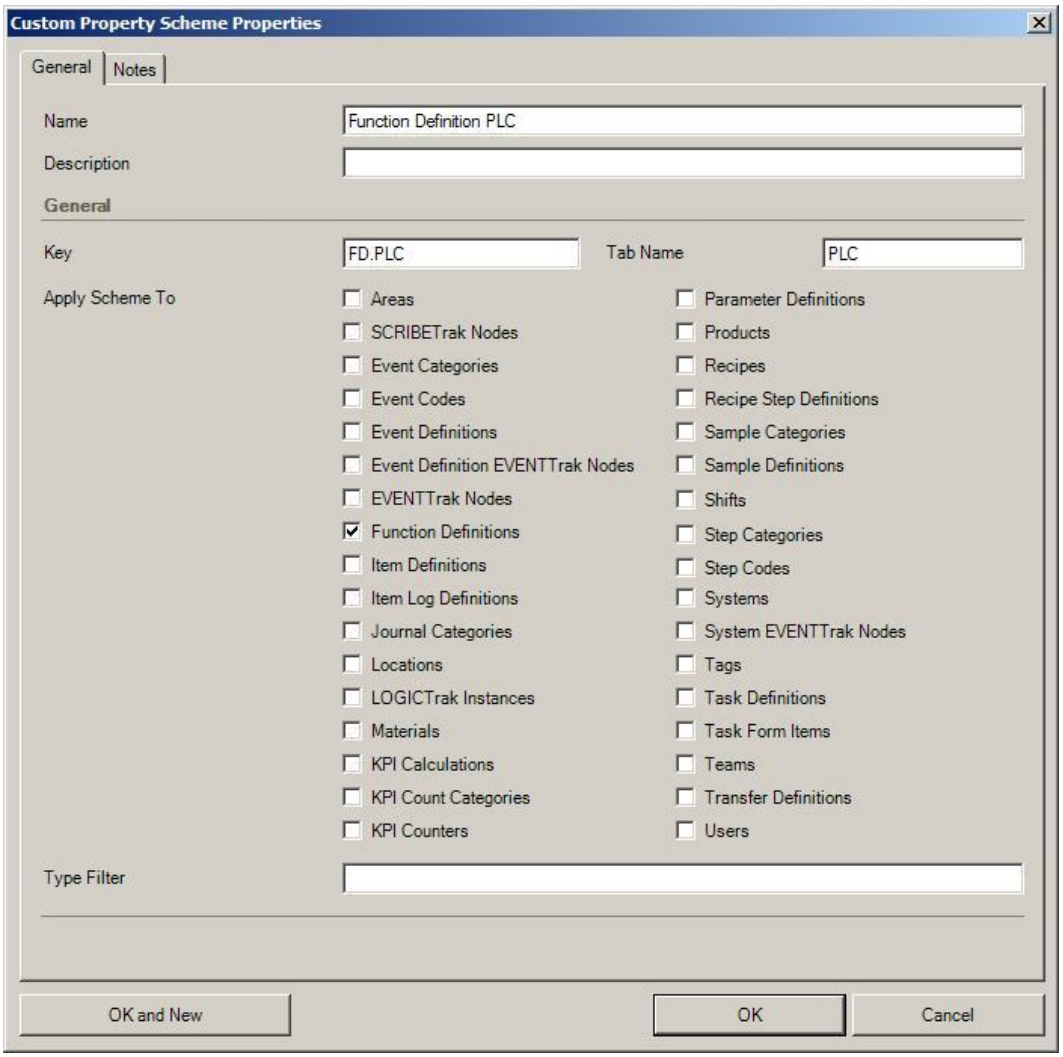
Assignment

Create a Custom Property Scheme for Function Definitions to define a State value used for triggering specific Functions. This Custom Property will be used with a custom EVENTTrak page in WEBTrak to manually trigger and execute the Steps of a Batch Recipe. Utilize existing WEBTrak dashboards to start, execute, and end a complete batch based on the configuration in MODELTrak. Add a dashboard that will display all Function Definitions and their corresponding State values using a custom query against the TrakSYS database.

Instructions

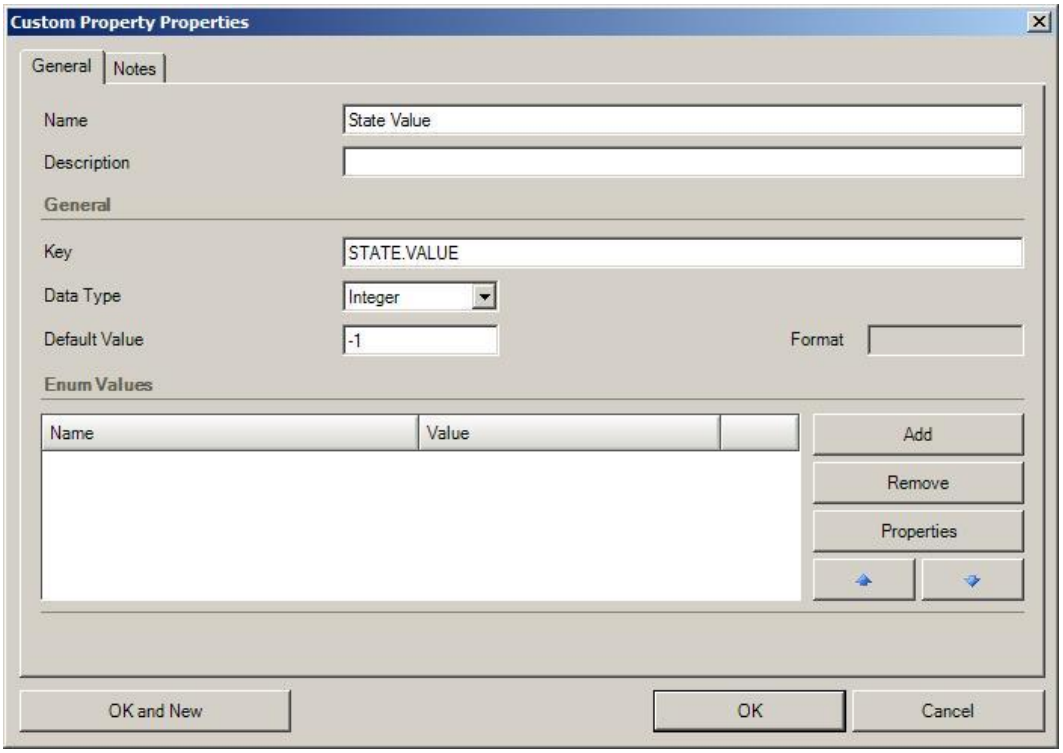
1. Open **MODELTrak** and select the **Custom Properties** panel.
2. Add a **New Custom Property Scheme** named **Function Definition PLC** with the following properties:
   a. General

Property	Value
Key	FD.PLC
Tab Name	PLC
Apply Scheme To	Function Definitions



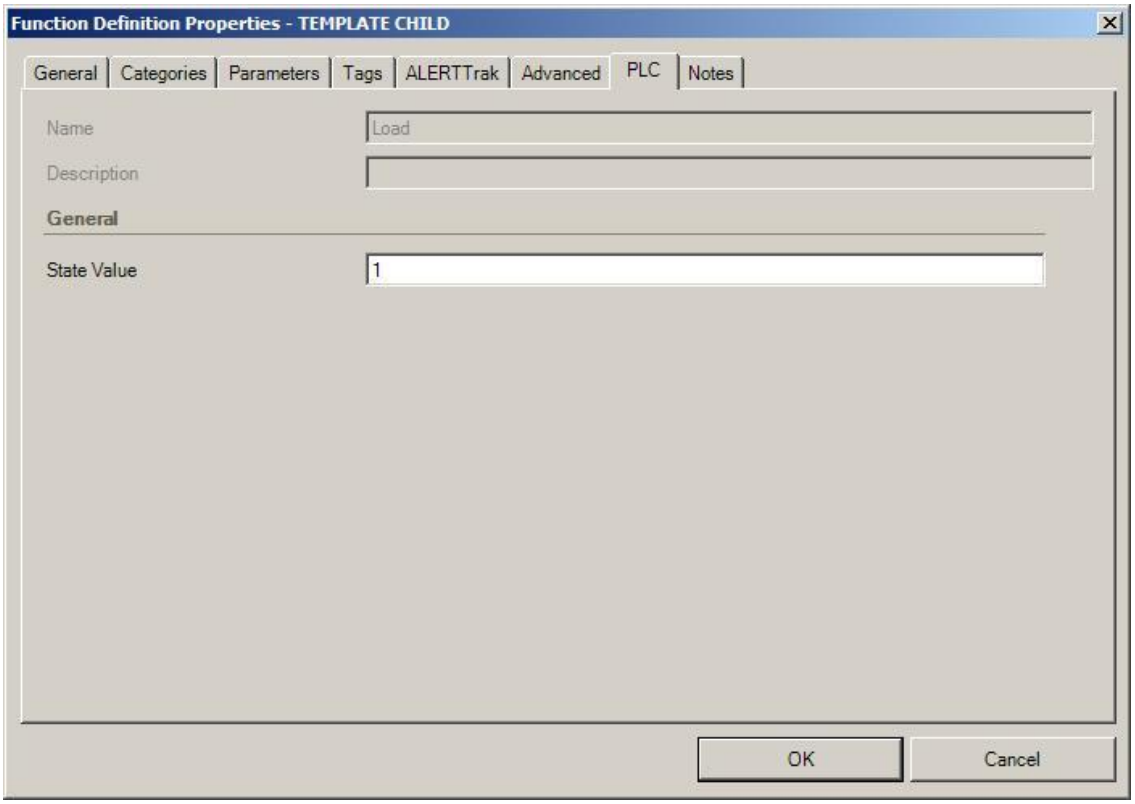
3. Add a **New Custom Property Group** named **General** to the **Function Definition PLC** Custom Property Scheme.
4. Add a **New Custom Property** named **State Value** to the **General** Group of the **Function Definition PLC** Scheme with the following properties:
   a. **General**

Property	Value
Key	STATE.VALUE
Data Type	Integer
Default Value	-1



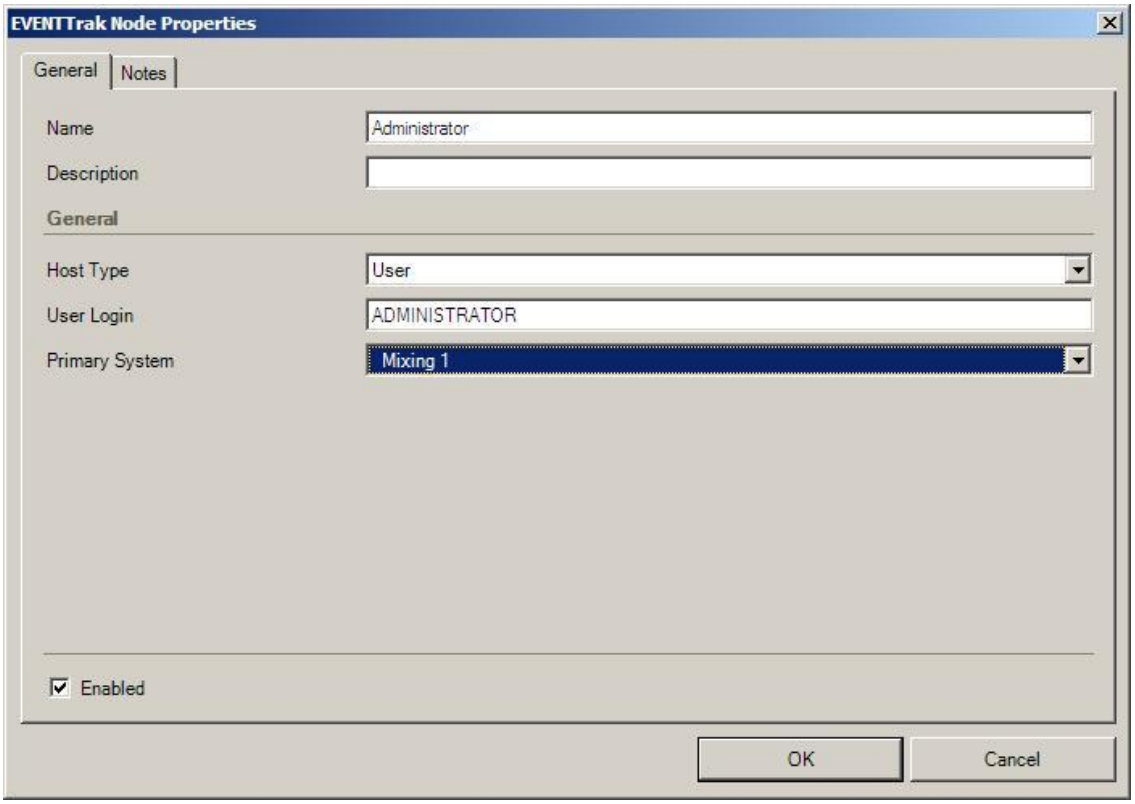
5. Go to the **Areas and Systems** panel and assign the following **PLC State Values** to the corresponding **Function Definitions** assigned to the **Mixing 1** System. Value assignments are made on the **PLC** tab of the **Function Definition Properties** dialog:

Sub-System [Unit]	Function Definition	State Value
Mixer	Load	1
Mixer	Mix	2
Mixer	Empty	3
Mixer	Clean	4

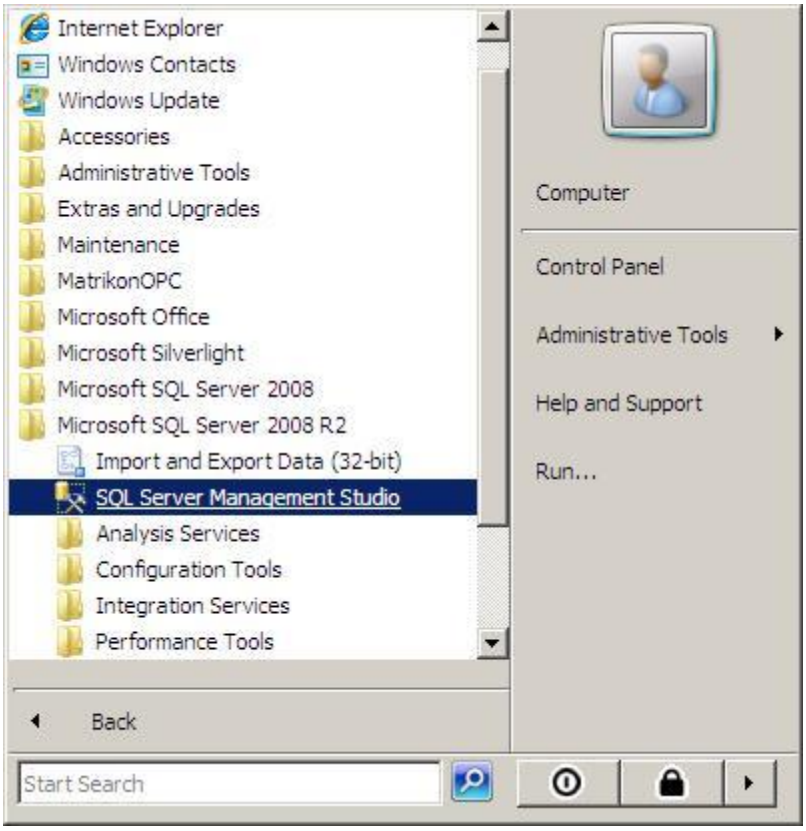


6. Go to the **Applications** panel and update the following properties for the **EVENTTrak Node** named **Administrator**:

Property	Value
Primary System	Mixing 1



7. Restart the **LOGICTrak Service** for the configuration changes to take effect.
8. Open **SQL Server Management Studio** and connect to the **(local)** Server using **Windows Authentication**.

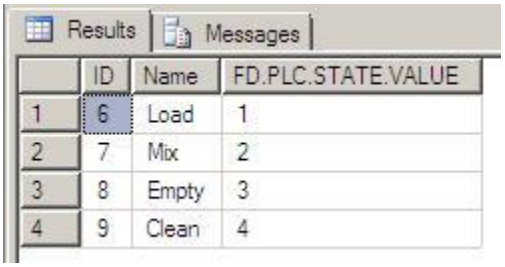


9. Open a **New Query** window for the **EDB_AT** database.
10. Write and execute a query in the New Query window that retrieves all Custom Properties for Function Definitions defined in MODELTrak.
    a. The query should return the following fields from database view **vwCustomPropertyFunctionDefinition**:
        i. ID

ii. Name
iii. FD.PLC.STATE.VALUE

b. The query should **join** the database view **vwCustomPropertyFunctionDefinition** with the table **tFunctionDefinition** using the fields named ID from each entity
   i. vwCustomPropertyFunctionDefinition.ID = tFunctionDefinition.ID

c. The query should be filtered to only return records where the field **IsTemplate** in **tFunctionDefinition** is equal to **0**.
   i. tFunctionDefinition.IsTemplate = 0


Results Messages

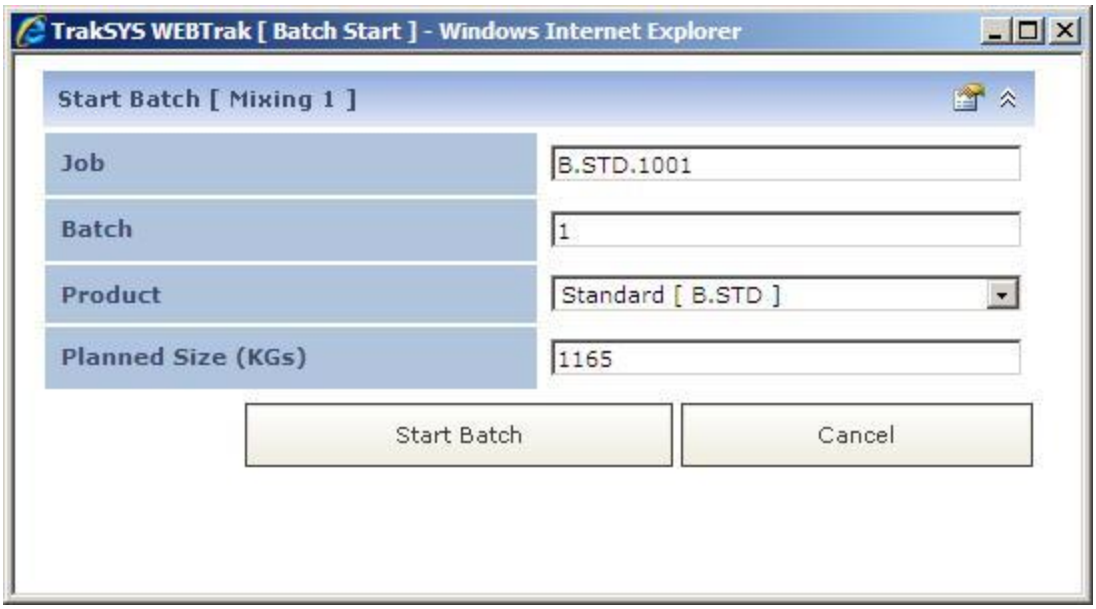
	ID	Name	FD.PLC.STATE.VALUE
1	6	Load	1
2	7	Mix	2
3	8	Empty	3
4	9	Clean	4

11. Open **WEBTrak** and login with the following account:

Login	administrator
Password	sa

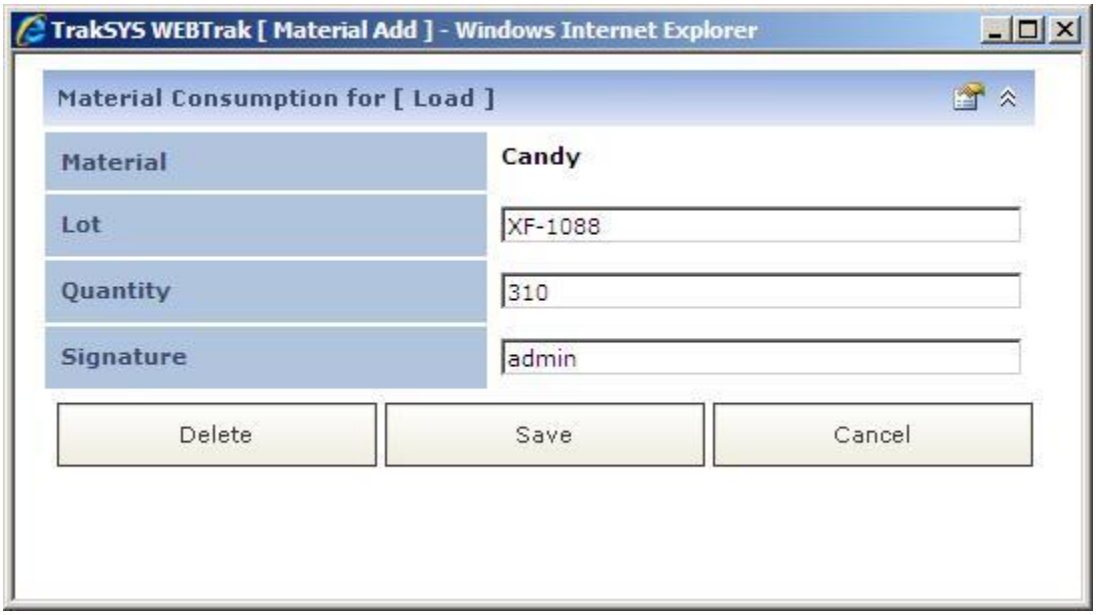
12. Navigate to the Report Group **TrakSYS Reports** | **TrakSYS Advanced Training** | **Lab: Batch Interfaces**.
13. Open the **Batching User Interface** dashboard.
14. Click the **Start Batch** button on the bottom left of the dashboard to initiate a new Batch for the Mixing 1 System. Enter the following information for the new Batch:

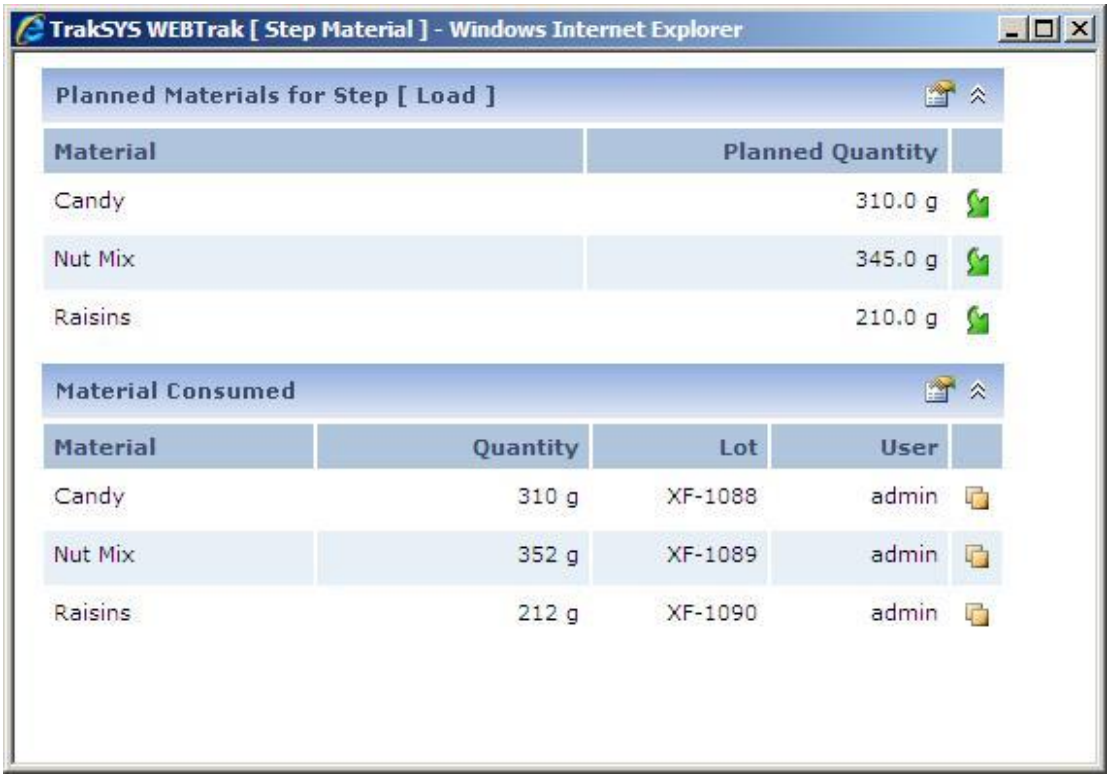
Job	B.STD.1001
Batch	1
Product	Standard [B.STD]
Planned Size (g)	1165



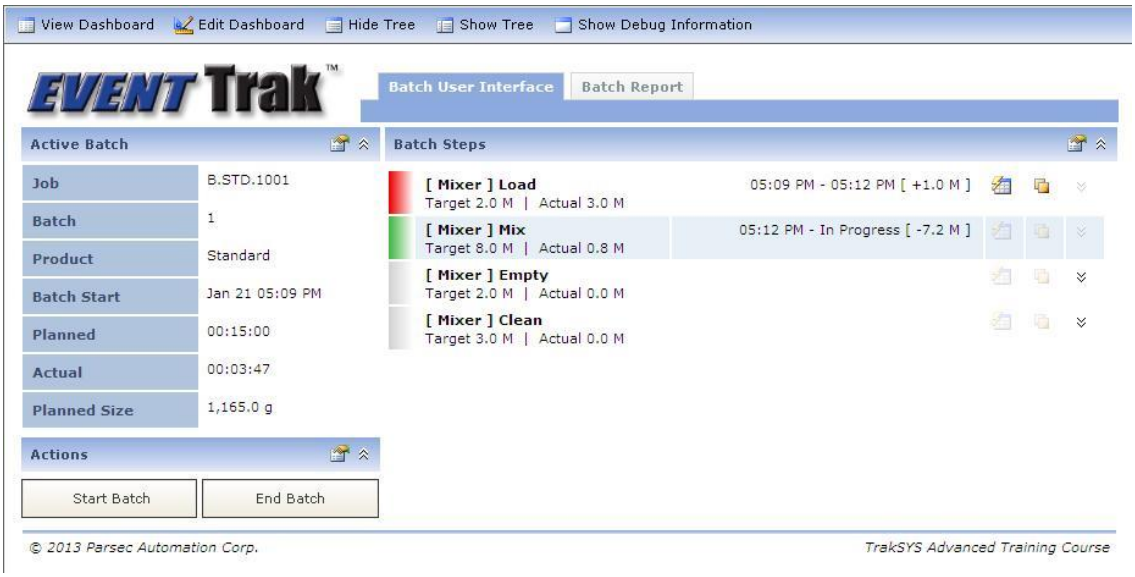
15. After the new Batch is loaded into the User Interface, click the icon for the **[Mixer] Load** step to open the **Step Material** entry form.
16. Click the icon for each entry in the **Planned Materials** list to enter the following **Material Consumption** data for the Batch:

Material	Lot	Quantity	Signature
Candy	XF-1088	310.0	admin
Nut Mix	XF-1089	351.6	admin
Raisins	XF-1090	211.7	admin



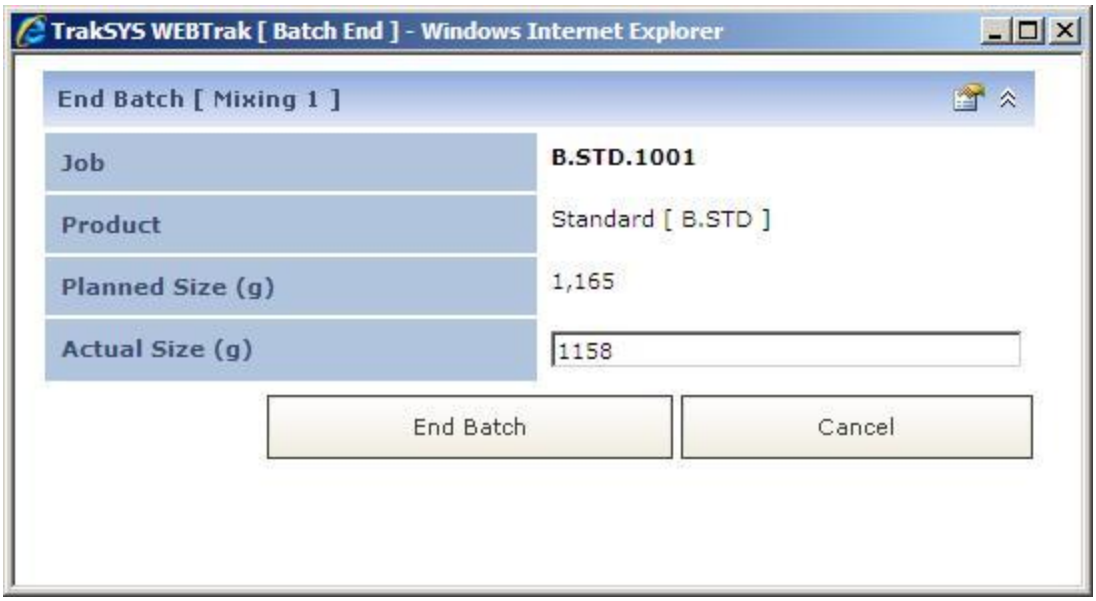


17. Click the icon for the **[Mixer] Mix** step after entering all **Material Consumption** data to proceed to the next step. Repeat this for the remaining steps in the Batch.



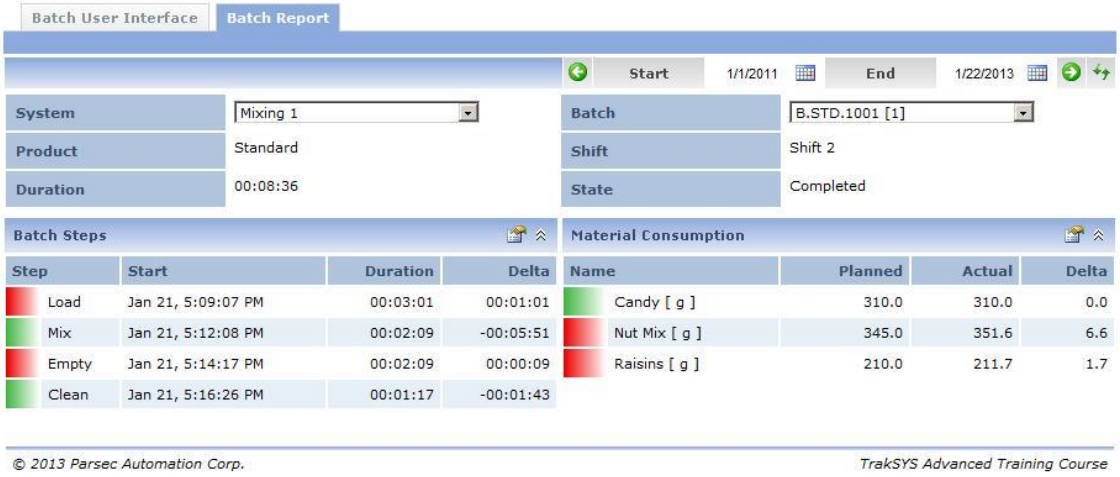
18. Click the **End Batch** button on the bottom left of the dashboard to end the current Batch for the Mixing 1 System. Enter the following information into the End Batch form:

Actual Size (g)	1158

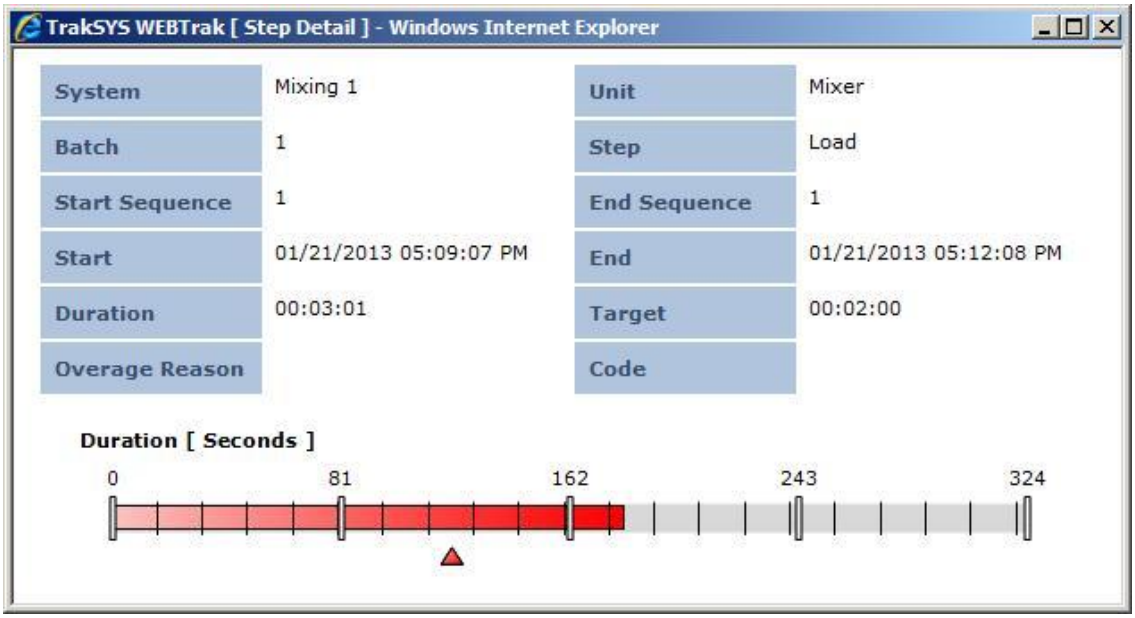


19. Open the **Batching Report** dashboard.
20. Select the following values from the user filters at the top of the dashboard to view data for the previously executed Batch:

System	Mixing 1
Batch	B.STD.1001 [1]



21. Click an entry in the Batch Steps or Material Consumption lists to view a drill down report with details from the selection.

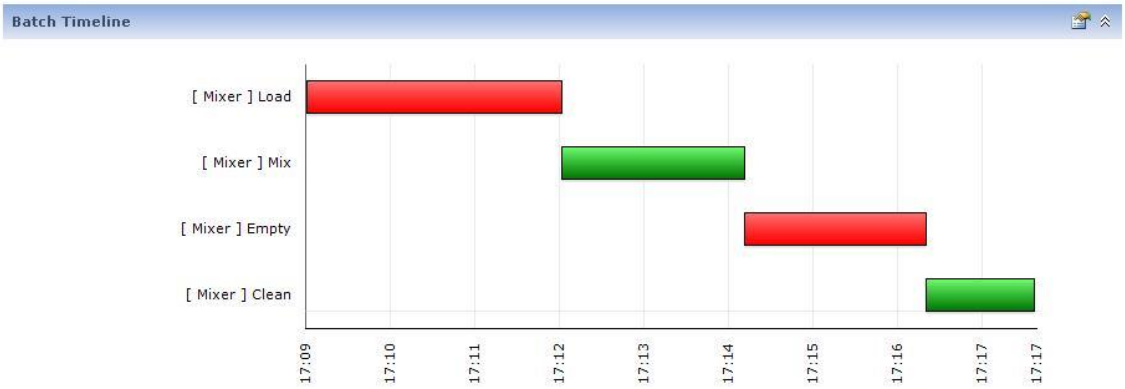

TrakSYS WEBTrak [Step Detail] - Windows Internet Explorer
System
Mixing 1
Unit
Mixer
Batch
1
Step
Load
Start Sequence
1
End Sequence
1
Start
01/21/2013 05:09:07 PM
End
01/21/2013 05:12:08 PM
Duration
00:03:01
Target
00:02:00
Overage Reason
Code
Duration [Seconds]
0
81
162
243
324

Advanced Assignment

Modify the **Source SQL Query** for the hidden Event State Chart named **Batch Timeline** on the **Batch Report** dashboard to display all of the steps for the selected System and Batch. Each bar of the chart should display steps for a specific Function Definition, with each slice representing a specific Recipe Step.

Tips and Hints

1. The web part is hidden on the dashboard. You will need to click the **Edit Dashboard** button to access it.
2. The Source SQL Query already has the parameters named **@SystemID**, **@BatchID**, **@StartDT**, and **@EndDT** defined, which correspond to the System, Batch, Start and End Date Times selected by the report user.
3. The query should be written against the database view **vwBatchStep**.
4. The format for the query can be found in the document **TrakSYS 8.0 Web Part Data Interface.pdf** posted on the TrakSYS Support Site. The relevant section for the Event State Chart is reprinted below.
5. Use the fields **BatchID** and **BatchName** for the query's Series values.
6. Use the fields **FunctionDefinitionID** and **FunctionDefinitionName** for the query's Group values.
7. The Slice ID for the query should be based on the field **BatchStepID**.
8. The Slice Name and Legend fields can both use the field **FunctionDefinitionName**.
9. Base the Slice Color on whether the Batch Step duration exceeded the expected running time.
   a. Example:
   ```
   CASE WHEN BatchStepDurationSecondsOverage > 0 THEN 'Red' ELSE
   'Green' END AS [SliceColor],
   ```
10. The query should be filtered with a **WHERE** clause by the Batch ID selected by the report user.
    a. Example:
    ```
    WHERE (BatchID = @BatchID)
    ```
11. Sort the query with an **ORDER BY** clause with the field **BatchStepStartSequence** in descending order.
12. The chart must be made visible by expanding/restoring the body of the web part and by (optionally) displaying the web part's Title.



Event State Chart

Each row in the following data table represents a single Slice on one row of the Event State Chart.

Required Columns

Column Name	Data Type	Description
SeriesID	INTEGER	A Series is a grouping of horizontal rows (Groups) on the chart. This field holds a unique numeric ID for the Series that this Slice belongs in. If no Series is required or applicable, return the value of -1 in this column.
SeriesName	STRING	A Series is a grouping of horizontal rows (Groups) on the chart. This field holds a unique string label for the Series that this Slice belongs in. If no Series is required or applicable, return an empty string in this column.
GroupID	INTEGER	A Group is a collection of Slices rendered on a single horizontal row on the chart. This field holds a unique numeric ID (within the parent Series) for the Group that this Slice belongs in. If no Group is required or applicable, return the value of -1 in this column.
GroupName	STRING	A Group is a collection of Slices rendered on a single horizontal row on the chart. This field holds a unique string label (within the parent Series) for the Group that this Slice belongs in. If no Group is required or applicable, return an empty string in this column.
SliceID	INTEGER	A Slice is a single block within one Group on the chart. This field holds a unique numeric ID (within the parent Group) for the Slice.
SliceName	STRING	A Slice is a single block within one Group on the chart. This field holds a string label for the Slice. This label is typically displayed in the ToolTip when the mouse hovers over the Slice on the chart.
SliceLegend	STRING	This is a string label for the Slice that is used for display in the Legend. This label may be the same or different than the SliceName. Legend items will be created for each unique value in the SliceLegend field over the entire data table,
SliceColor	STRING	This is the color of the Slice (color name, web color value, or integer RGB color value).
SliceStartDateTime	DATETIME	This is the start date and time of the Slice.
SliceEndDateTime	DATETIME	This is the end date and time of the Slice.

Optional Columns

Column Name	Data Type	Description
RangeStartDateTime	DATETIME	Coupled with the RangeEndDateTime, when these optional columns are included, they override the default chart behavior and control the start and end points of the date/time X axis scale.

RangeEndDateTime	DATETIME	Coupled with the RangeStartDateTime, when these optional columns are included, they override the default chart behavior and control the start and end points of the date/time X axis scale.

Sample Data

SeriesID	SeriesName	GroupID	GroupName	SliceID	SliceName	SliceLegend	SliceColor	SliceStartDateTime	SliceEndDateTime
3	Casepacker	1		1	Starved	Starved	#BDBDF2	1/1/2008 09:41:00	1/1/2008 09:51:00
3	Casepacker	1		1	Starved	Starved	#BDBDF2	1/1/2008 10:16:00	1/1/2008 09:51:00
3	Casepacker	1		1	Starved	Starved	#BDBDF2	1/1/2008 12:02:00	1/1/2008 10:27:00
2	Labeler	5	Backed Up	1	Backed Up	Backed Up	Orange	1/1/2008 16:04:00	1/1/2008 16:22:00
2	Labeler	4	Side Cover Open	1	Active	Fault	Red	1/1/2008 09:38:00	1/1/2008 16:29:00

Advanced Scripting

Assignment

Create an Advanced Script Class in MODELTrak that implements various business rules and logic for Packaging Fillers. The business rules will return true or false values based on specific equipment Tag values that will be used as inputs. The Script Class functions will be called from Script Tags that will be added for the Filler on Packaging Line 1.

Instructions

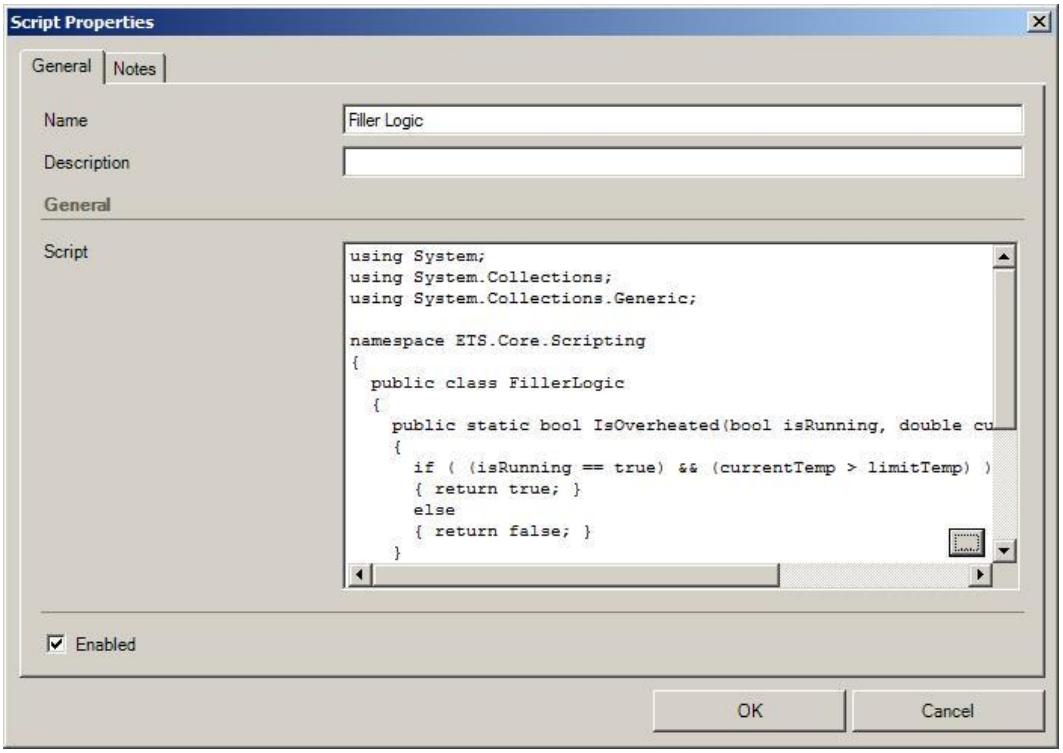
1. Open **MODELTrak** and select the **Administration** panel.
2. Create a **New Standard Script** named **Filler Logic** in the Script Library's **Training** folder. Edit the **Script** property to create a C#.NET class with the following properties and methods:
   a. Change the name of the class from **TODO** to **FillerLogic**.
   b. Add the following methods:

Name	IsOverheated
Method Type	public static
Return Data Type	bool
Parameters	isRunning : bool currentTemp : double limitTemp : double

Method Description	Returns a true or false value based on the following logical condition:
	
	`(isRunning = true) and`
	`(currentTemp > limitTemp)`

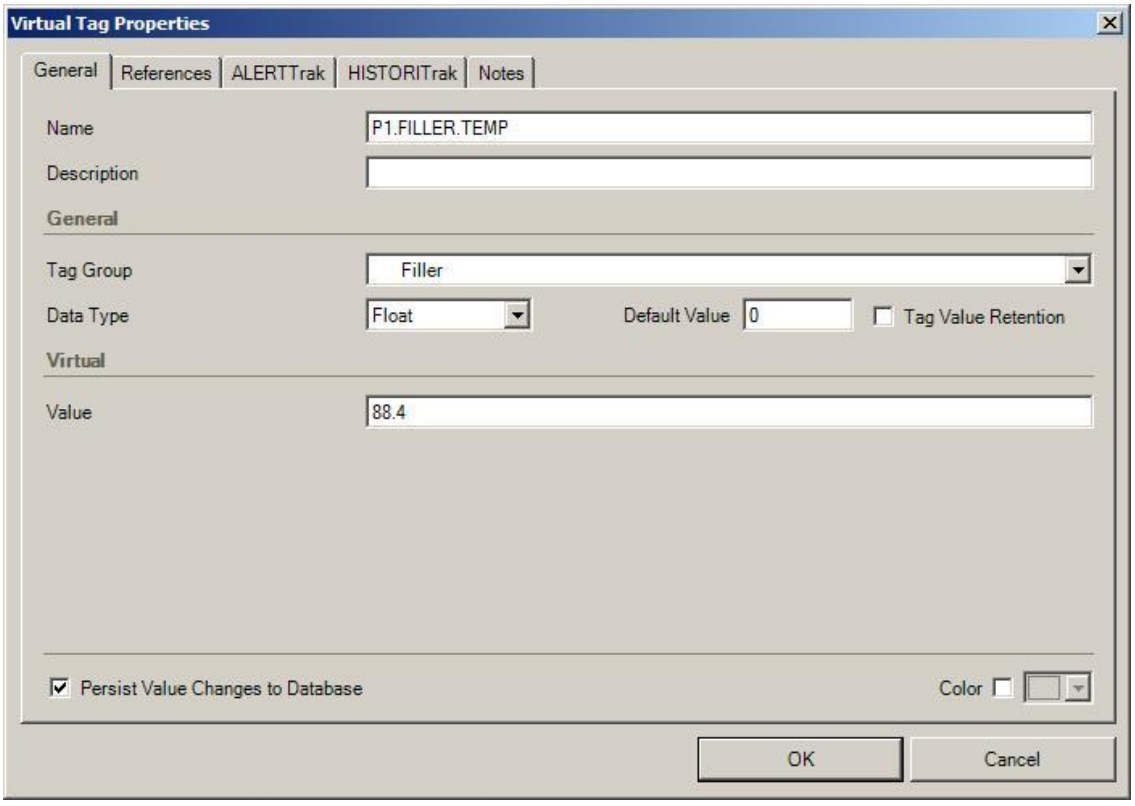
Name	IsIdle
Method Type	public static
Return Data Type	bool
Parameters	isRunning : bool notScheduled : bool productCode : string

Method Description	Returns a true or false value based on the following logical condition:
	
	`(isRunning = true) and`
	`(notScheduled = false) and`
	`(productCode = "")`



3. Go to the **Tags** panel and create the following **New Virtual Tags** in the Filler Tag Group for Packaging 1 (**OCF\Packaging\Packaging 1\Filler**):

Name	Data Type	Value
P1.FILLER.RUNNING	Discrete	1
P1.FILLER.TEMP	Float	88.4



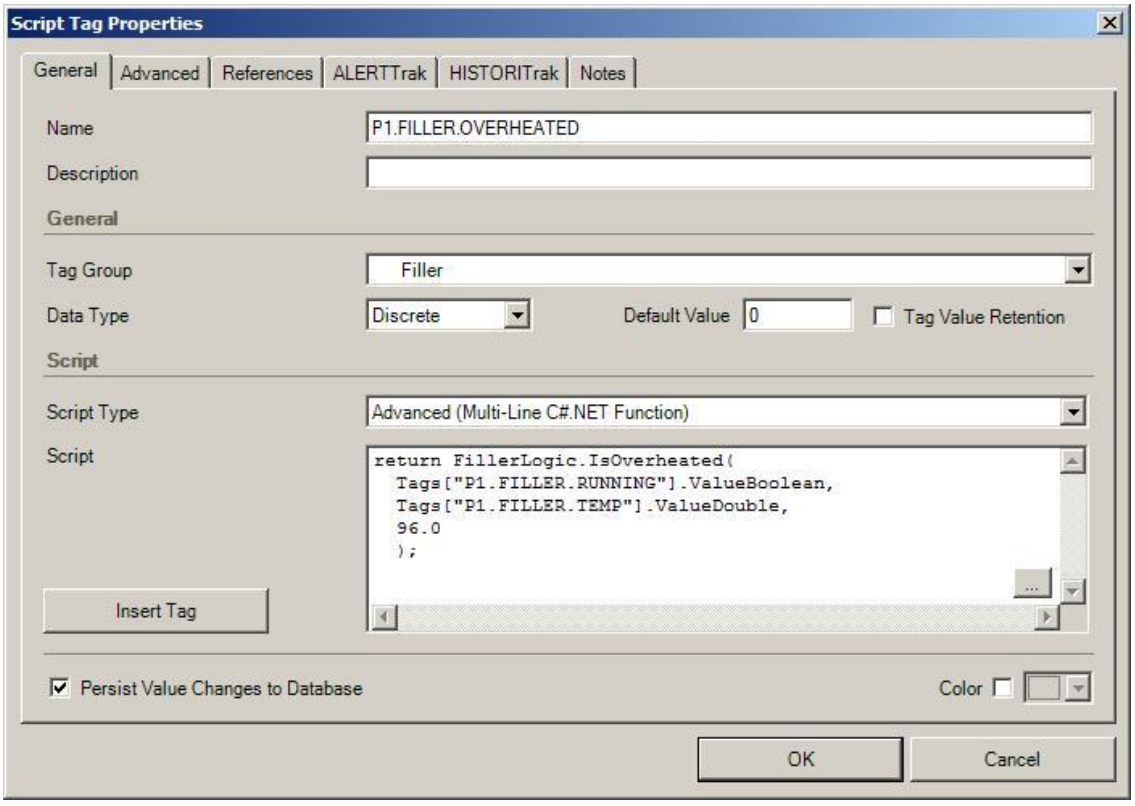
4. Create the following **New Script Tags** in the Filler Tag Group for Packaging 1:

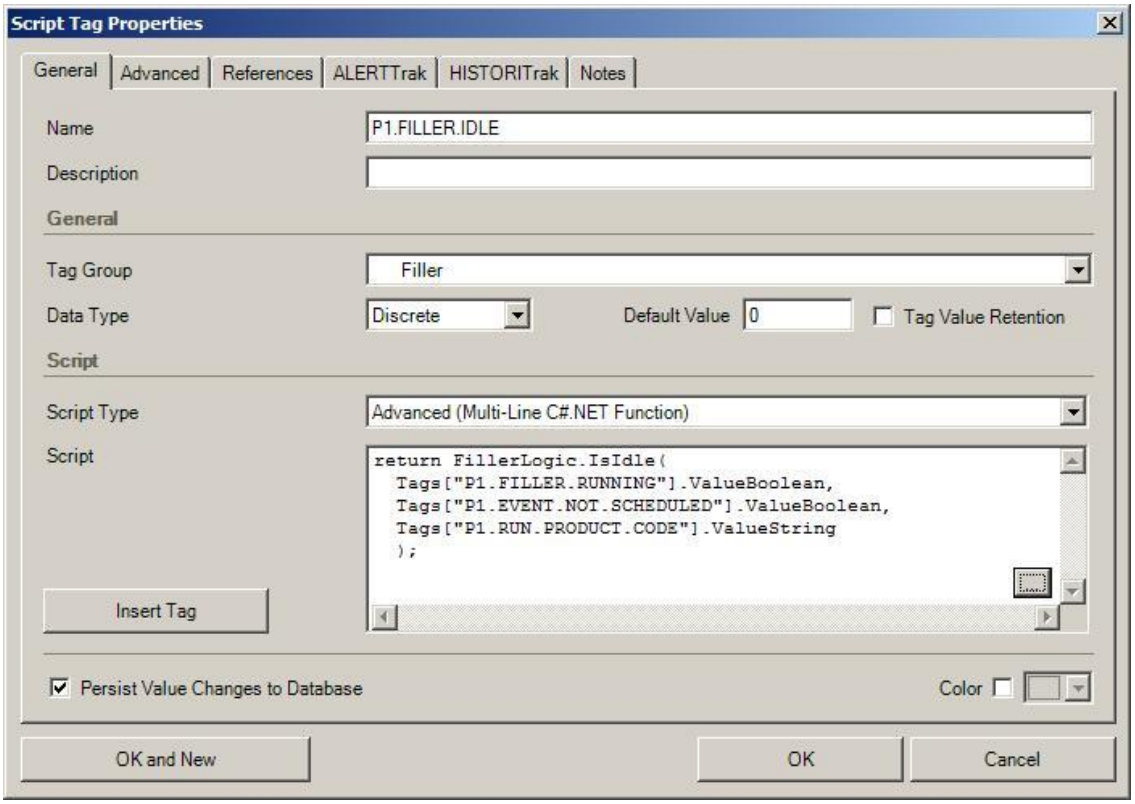
Name	PL1.FILLER.OVERHEATED
Data Type	Discrete
Script Type	Advanced (Multi-Line C#.NET Function)

Script	`return FillerLogic.IsOverheated(`
	`  Tags["P1.FILLER.RUNNING"].ValueBoolean,`
	`  Tags["P1.FILLER.TEMP"].ValueDouble,`
	`  96.0`
	`  );`

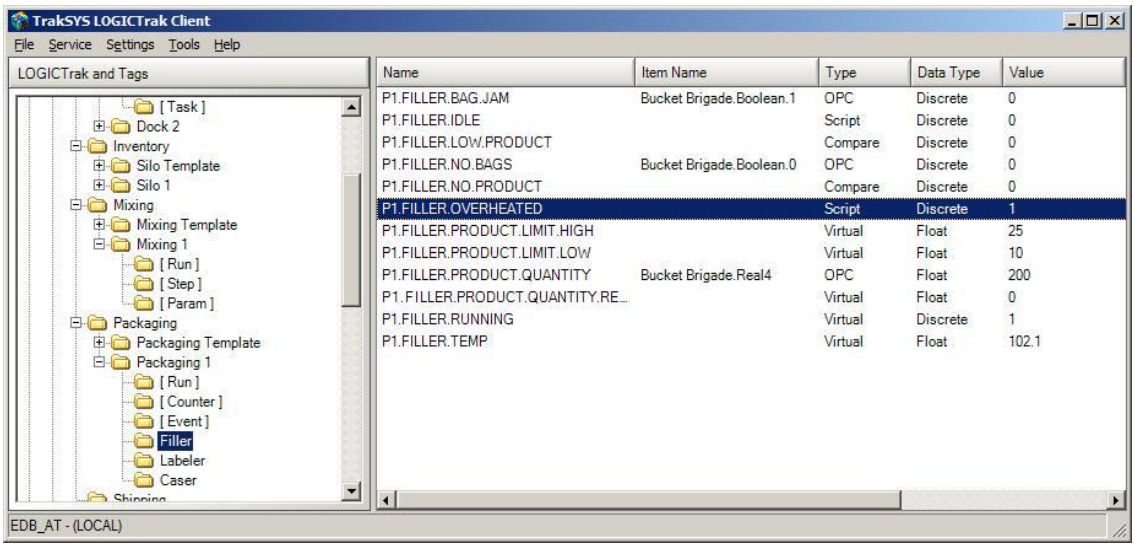
Name	PL1.FILLER.IDLE
Data Type	Discrete
Script Type	Advanced (Multi-Line C#.NET Function)

Script	`return FillerLogic.IsIdle(`
	`  Tags["P1.FILLER.RUNNING"].ValueBoolean,`
	`  Tags["P1.EVENT.NOT.SCHEDULED"].ValueBoolean,`
	`  Tags["P1.RUN.PRODUCT.CODE"].ValueString`
	`  );`





5. Open the **LOGICTrak Client** and restart the **LOGICTrak Service** for the configuration changes to take effect.
6. Navigate to the Filler Tag Group for Packaging 1.
7. Change the value for tag **P1.FILLER.TEMP** from **88.4** to **102.1**. Confirm that the value for tag **P1.FILLER.OVERHEATED** changes from 0 to 1. Change the value for P1.FILLER.TEMP back to 88.4.
8. Navigate to the [Run] Tag Group. Change the value for tag **P1.RUN.PRODUCT.CODE** to a blank value (empty string). Return to the Filler Tag Group and confirm that the value for tag **P1.FILLER.IDLE** is set to **1**. Change the value for P1.RUN.PRODUCT.CODE back to P.STD and confirm that P1.FILLER.IDLE resets to 0.



Advanced Assignment

Create a System Entity Script Class that automatically writes a comment into the Notes for all Events triggered under a System. The Script Class will update the Event either by executing a query directly against the TrakSYS database or by using a Data Model class from the TrakSYS API to load and update the Event's data. Results may be reviewed using the standard Event Summary reports in WEBTrak.

Tips and Hints

1. Add code for the Event update to the method **PostScanEventStart(context)** **OR** the method **PostScanEventEnd(context)** in the template for the System Script Class.
2. The ID for the triggered Event can be retrieved using **context.EventID** from the method's context object parameter.
3. Queries may be executed directly against the TrakSYS database using the method **context.Execute(sql)**.
4. Event data is stored in the table **tEvent** of the TrakSYS database. The fields **ID** and **Notes** would be used in an update query.
5. An instance of the TrakSYS API service can be accessed using **context.Api** from the method's context object.
6. The TrakSYS knowledgebase article entitled Updating an Entity in the Database provides an example of using the TrakSYS API to update an existing Event in the database. It is reprinted below as a reference.
7. Assign the **Script Class Name** to the **Packaging 1** System. The assignment can be made under the **Advanced** tab of the System Properties dialog. Be sure to assign the Class name rather than the Script name to the System.
8. Restart the **LOGICTrak Service** for the configuration changes to take effect prior to triggering new Events.

Updating an Entity in the Database

Applies To | TrakSYS 7.X | TrakSYS 8.X

Summary

This sample includes the code needed to update a TrakSYS entity in the database (an Event record in this example).

Sample

```
// create a reference to the api service
ETS.Core.Api.ApiService api = new ETS.Core.Api.ApiService();

// create a model object to hold the results of the load
ETS.Core.Api.Models.Data.DbEvent ev;

// load the entity with ID 123 from the database
ev = api.Data.DbEvent.Load.ByID(123);

// modify the properties of the model object as needed
ev.Notes = "new notes have been added";
// etc...

// create a result object to determine the success of the operation
ETS.Core.Api.Models.Result<ETS.Core.Api.Models.Data.DbEvent> result;

// update the entity in the database
result = api.Data.DbEvent.Save.UpdateExisting(ev);

// examine the results of the operation
if (result.Success)
{
  // success code
}
else
{
  // failure code
}
```

INTELLITrak

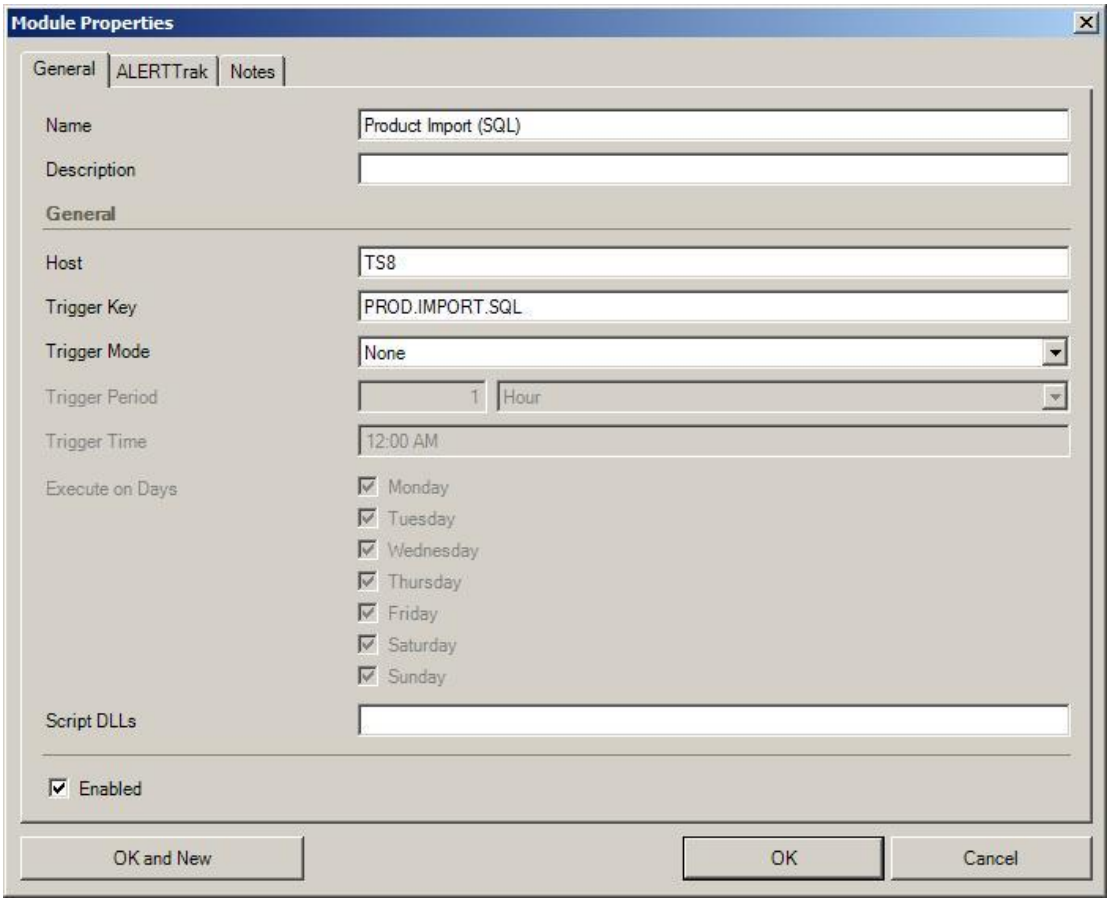
Assignment

Create an INTELLITrak module that imports Product data from an Excel file and updates information in an existing PRODUCTTrak Product Set. Export the updated Product Set to an XML file for review and further processing by an external business system.

Instructions

1. Open **MODELTrak** and select the **INTELLITrak** panel.
2. Create a new **Module** named **Product Import (SQL)** with the following properties:

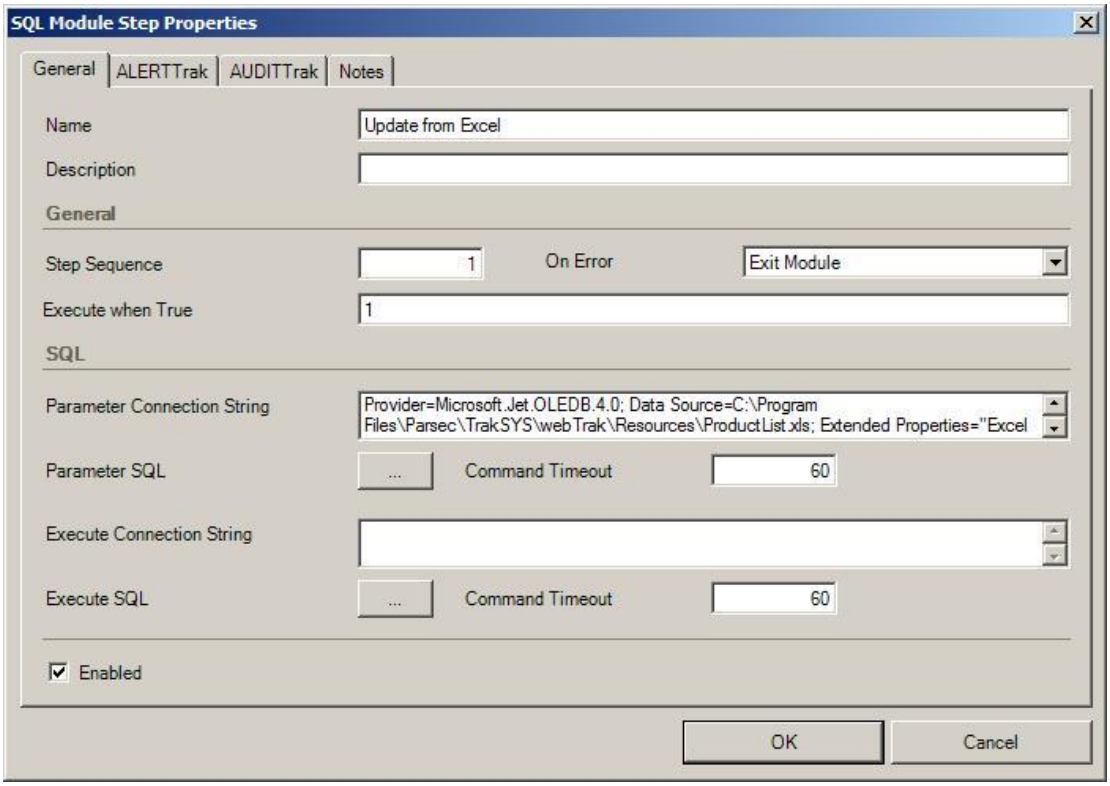
Property	Value
Host	TS8
Trigger Key	PROD.IMPORT.SQL
Trigger Mode	None



3. Add a **New SQL Module Step** to the **Product Import (SQL)** Module with the following properties:

Name	Update from Excel
Step Sequence	1
On Error	Exit Module
Parameter Connection String	Provider=Microsoft.Jet.OLEDB.4.0; Data Source=C:\Program Files\Parsec\TrakSYS\webTrak\Resources\ProductList.xls; Extended Properties="Excel 8.0; HDR=Yes; IMEX=1";
Parameter SQL	`SELECT * FROM [ProductList$]`
Execute Connection String	

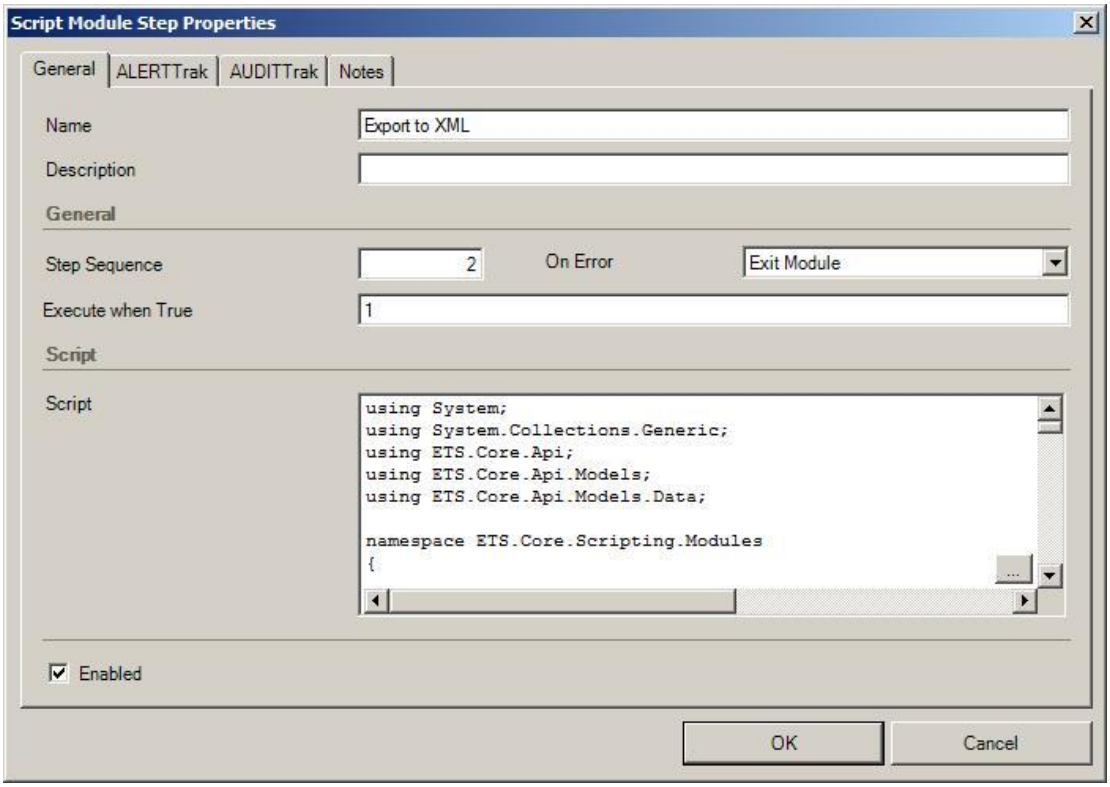
Execute SQL	`UPDATE tProduct`
	`SET`
	`  Attribute01 = '{param.sql|ValidatedRate}',`
	`  Attribute02 = '{param.sql|NominalRate}',`
	`  Attribute03 = '{param.sql|BagsPerCase}',`
	`  Attribute04 = '{param.sql|BarCode}'`
	`WHERE`
	`  (ProductCode = '{param|ID}')`



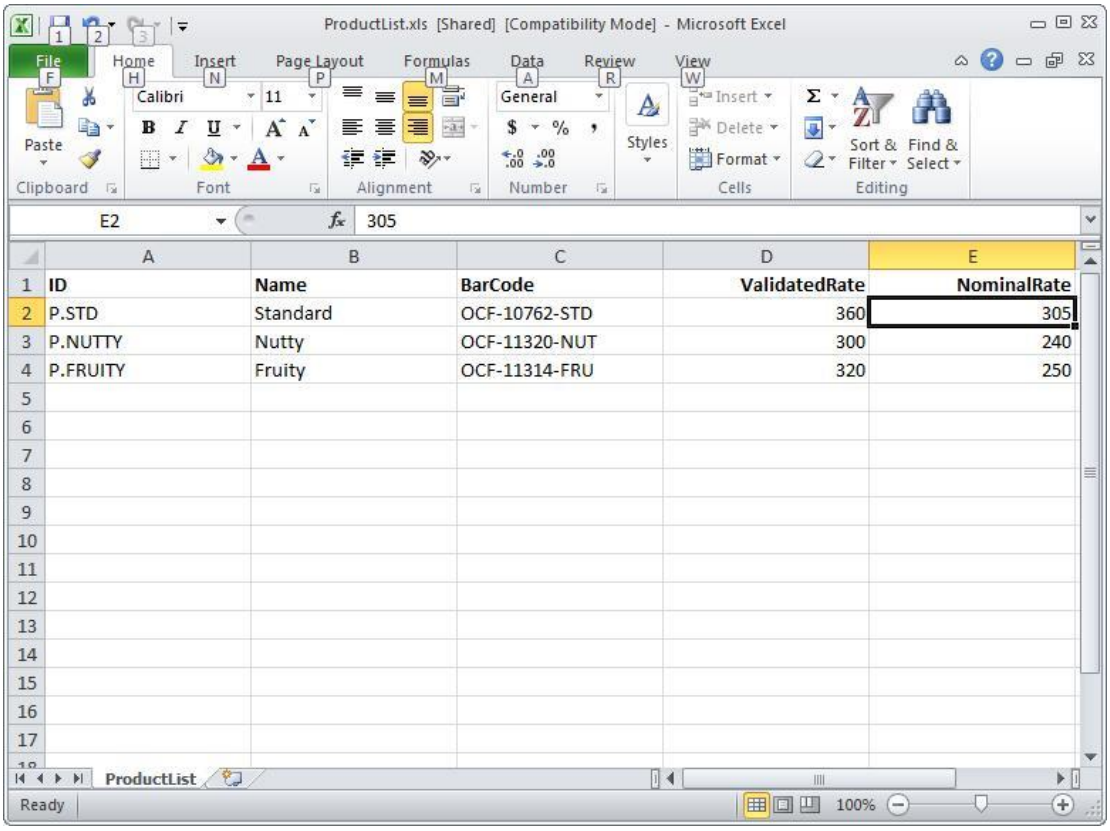
4. Add a **New Script Module Step** to the **Product Import (SQL)** Module with the following properties:

Name	Export to XML
Step Sequence	2
On Error	Exit Module
Script	*Add the following content to the **Execute()** method in the basic Script template:*

```
public override bool Execute(IModuleContext ctx)
{
  // set filename to output
  string filename = @"C:\Users\Administrator\Desktop\productdata.xml";
  try
  {
    // delete existing file
    if (System.IO.File.Exists(filename))
    { System.IO.File.Delete(filename); }

    // create new xml file
    using (System.Xml.XmlWriter writer =
System.Xml.XmlWriter.Create(filename))
    {
      // write root element
      writer.WriteStartDocument();
      writer.WriteStartElement("products");

      // retrieve list of products
      int productSetID = 3;
      List<DbProduct> productList =
ctx.Api.Data.ListOf.DbProducts.GetList.ForProductSetID(productSetID);
      foreach (DbProduct prod in productList)
      {
        // add product element
        writer.WriteStartElement("product");

        // add data elements
        writer.WriteElementString("product-code", prod.ProductCode);
        writer.WriteElementString("name", prod.Name);
        writer.WriteElementString("theoretical-rate", prod.Attribute01);
        writer.WriteElementString("standard-rate", prod.Attribute02);
        writer.WriteElementString("bags-per-case", prod.Attribute03);
        writer.WriteElementString("barcode", prod.Attribute04);

        writer.WriteEndElement();
      }

      // end document
      writer.WriteEndElement();
      writer.WriteEndDocument();
    }

    return true;
  }
  catch (Exception ex)
  {
    // log exception
    ctx.Api.Util.Log.WriteException(ex);
    return false;
  }
}
```

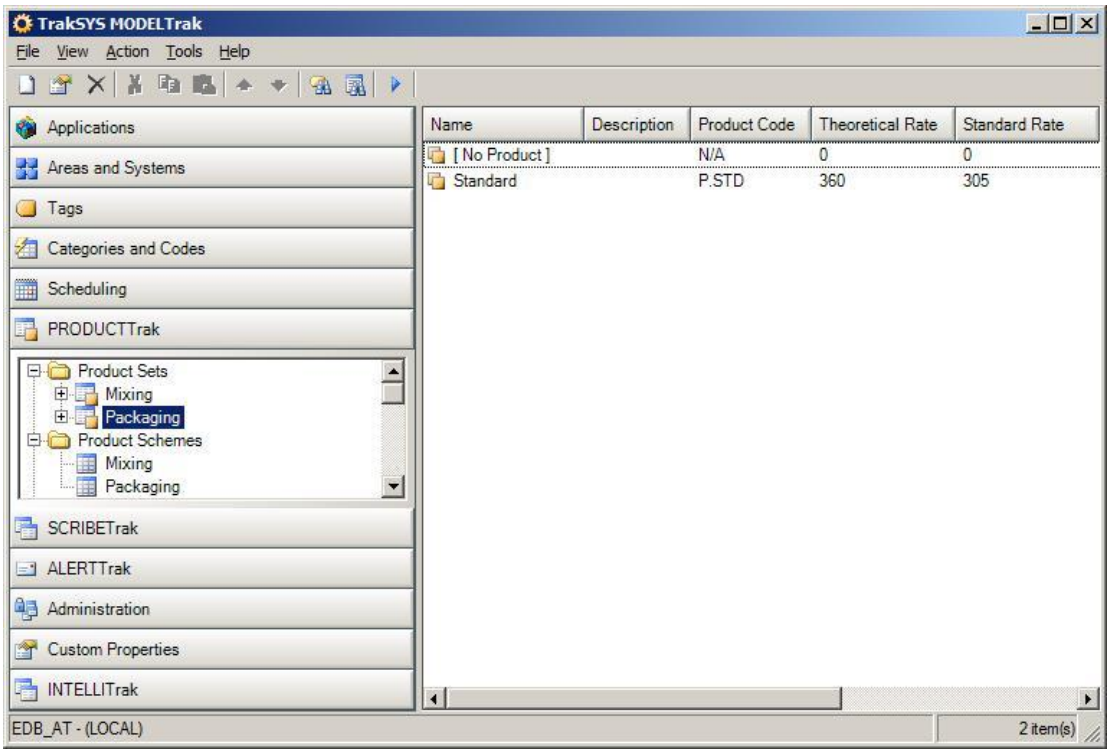


5. Open **LOGICTrak Client** and select the **INTELLITrak Modules** folder.
6. Right-click on the **Product Import (SQL)** Module to **Start** it.
7. Open the Excel file named ProductList.xls from the shortcut on the Desktop.
8. Change the **NominalRate** value for the **Standard** [**P.STD**] Product from **280** to **305. Save** and **Close** the updated file.



	A	B	C	D	E
1	**ID**	**Name**	**BarCode**	**ValidatedRate**	**NominalRate**
2	P.STD	Standard	OCF-10762-STD	360	305
3	P.NUTTY	Nutty	OCF-11320-NUT	300	240
4	P.FRUITY	Fruity	OCF-11314-FRU	320	250
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					

9. Return to the LOGICTrak Client and right-click on the **Product Import (SQL)** Module to **Execute** it manually.
10. Go to the **PRODUCTTrak** panel in **MODELTrak** and select the **Packaging** Product Set. Confirm that the **Standard Rate** for the **Standard** Product has been updated to 305.



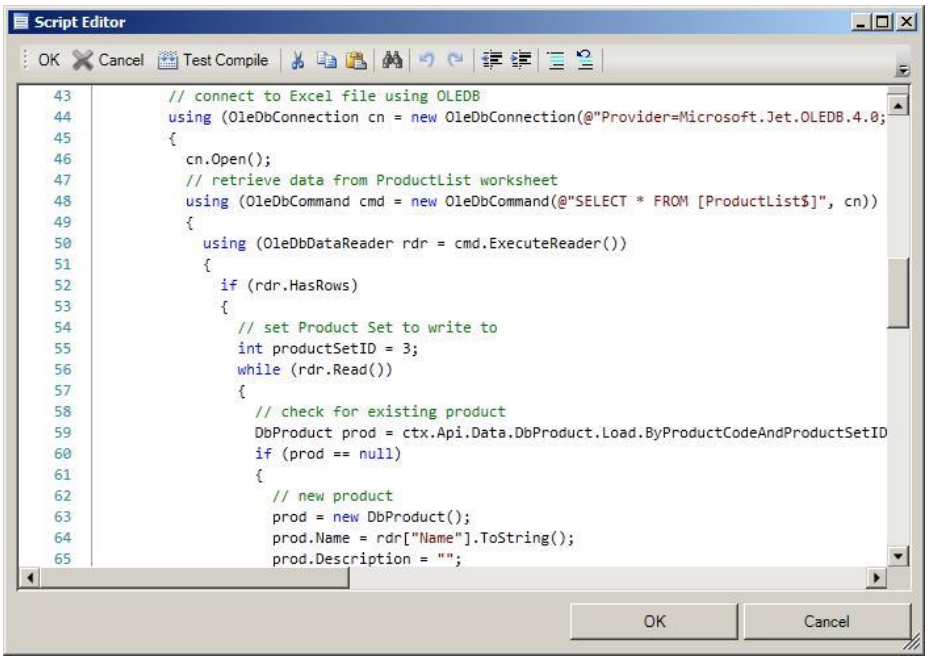
11. Navigate to the computer's **Desktop** and confirm that the file **productdata.xml** has been created. Open the file to confirm that the updated Product data was exported successfully.

Advanced Assignment

Modify the Script Module Step named Import from Excel for the **Product Import (Script)** Module to both insert new Products and update existing Products in the Packaging Product Set. The **Execute()** method in the Script Module Step already contains C#.NET code to open the Excel file using OLEDB and read the existing records into an **OleDbDataReader** object. Utilize the **TrakSYS API** to insert or update records in the TrakSYS database.

Tips and Hints

1. Attribute values in the OleDbDataReader object may accessed using the syntax `rdr["FieldName"].ToString()`.
   a. Field name values are based on the column headers found in the source Excel file.
   b. All attributes in the OleDbDataReader object may be accessed as strings.
2. Utilize the function `ctx.Api.Data.DbProduct.Save.InsertAsNew()` when creating new Product records in the database, and `ctx.Api.Data.DbProduct.Save.UpdateExisting()` for updating Product records.
3. Review the existing Execute SQL Parameter created earlier in the lab to understand how the fields in the Excel file map to the fields in the Product table.
4. The following fields/attributes are required when inserting a new Product into the database:
   a. **Name** – From Excel
   b. **Description** – Empty string
   c. **Notes** – Empty string
   d. **ProductSetID** – From existing variable named **productSetID**
   e. **ProductCode** – From Excel
   f. **VersionState** – Use the enum value **ETS.Core.Enums.VerisionState.Active**
   g. **Color** – Set to -1
   h. **Enabled** – Set to true
5. A possible solution is included as a reference named **Product Import (Script) (Complete)**.



Excel Reports

Assignment

Create Excel reports in WEBTrak that will export data from the TrakSYS database into a new or existing Excel workbook. The new files will be populated by custom queries executed against the TrakSYS database whenever the reports are run by an end user. One Excel report will export product data from an existing Product Set in MODELTrak, while a second report will merge Tag History data into an existing, pre-formatted Excel file.

Instructions

1. Open **WEBTrak** and login with the following account:

Login	administrator
Password	sa

2. Navigate to the Report Group **TrakSYS Reports** | **TrakSYS Advanced Training** | **Lab: Excel Reports**.
3. Add a new **Excel** report named **Product Export** to the Report Group with the following parameters:
   a. **Excel (Advanced)**

Key	TSST.EXCEL.PRODUCT

   b. **General**

Output File Name	ProductExport
Output Type	Excel 1997-2003 Workbook (xls)

   c. **Data 1**

SQL	`SELECT *`
	`FROM tProduct`
	`WHERE (ProductSetID = 3)`

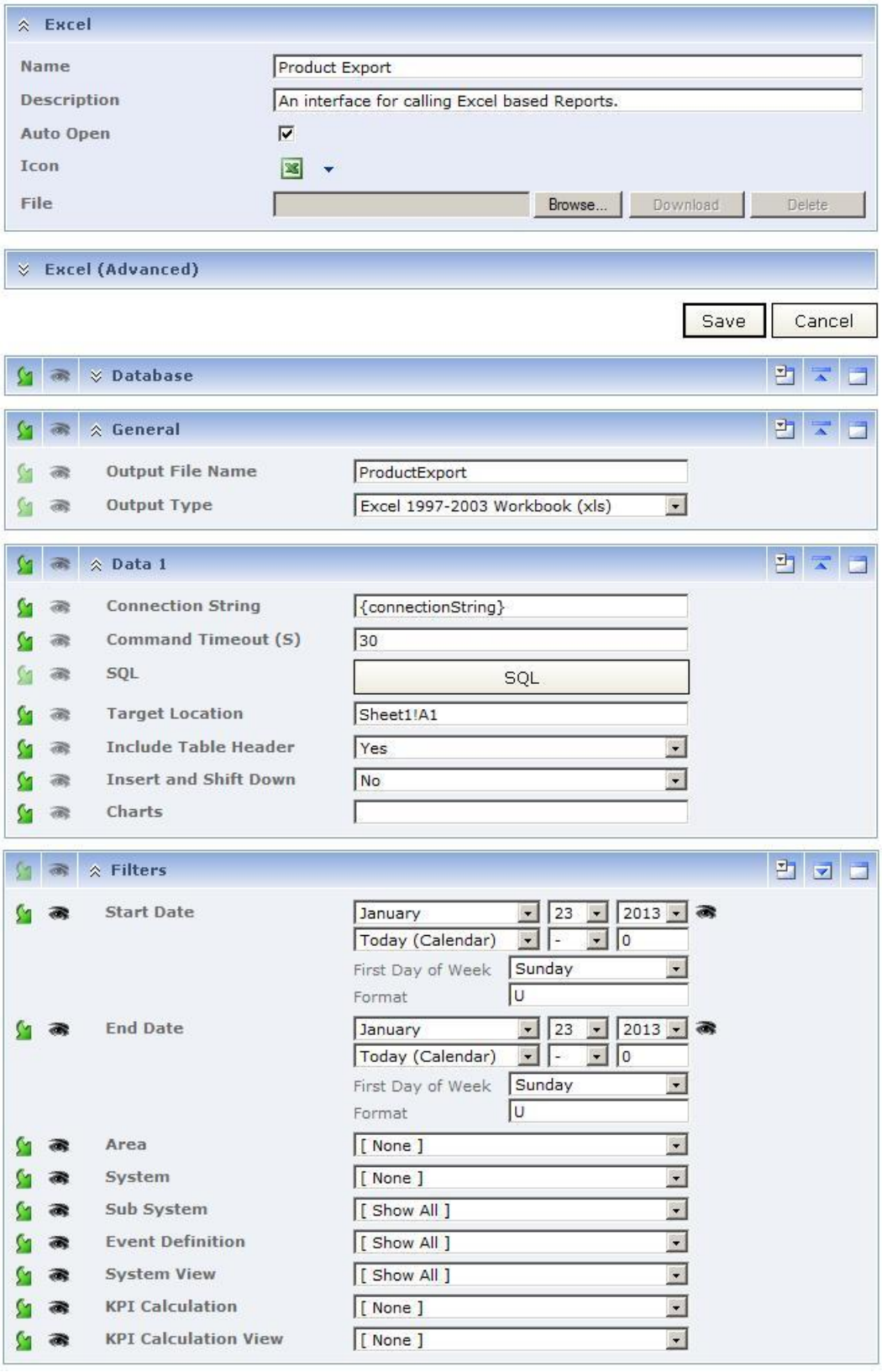
Target Location	Sheet1!A1
Include Table Header	Yes
Insert and Shift Down	No

   d. **Filters**

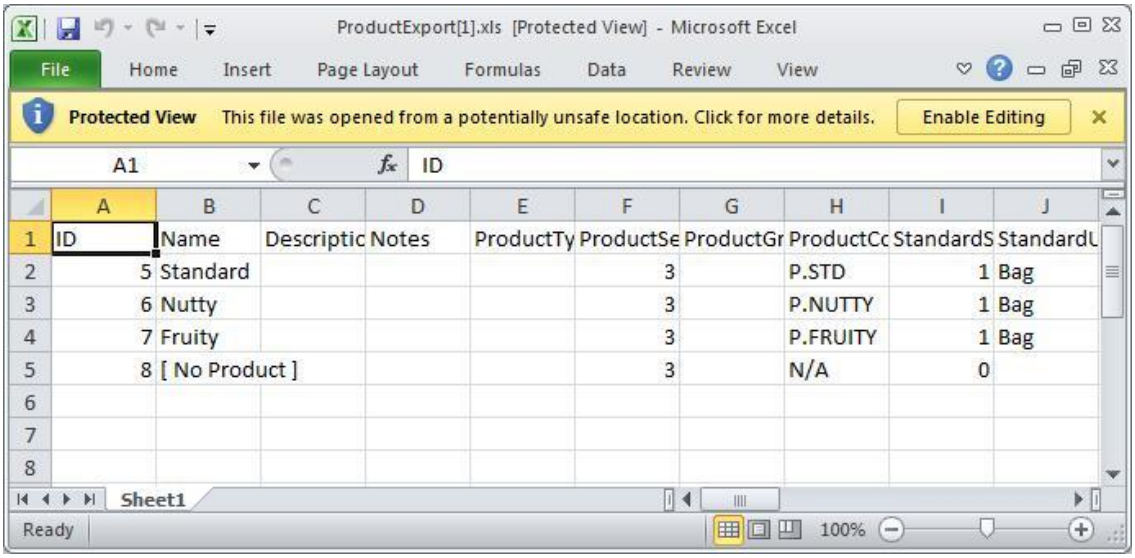
[Section Header]	Click the  icon in the section header to hide this section of parameters from the end user. The icon should be grayed-out after clicking it.

   a. **Excel** *[top of page]*

Auto Open	Checked



2. Open the new Product Export report in the WEBTrak tree view to export Product data into an Excel file.



ID	Name	Descriptic	Notes	ProductTy	ProductSe	ProductGr	ProductCc	StandardS	StandardU
5	Standard				3		P.STD	1	Bag
6	Nutty				3		P.NUTTY	1	Bag
7	Fruity				3		P.FRUITY	1	Bag
8	[No Product]				3		N/A	0	

3. Add a new **Excel** report named **Tag History** to the Report Group with the following parameters:
   a. **Excel (Advanced)**

Key	TSST.EXCEL.TAG

   b. **General**

Output File Name	TagHistory
Output Type	Excel 1997-2003 Workbook (xls)

   c. **Data 1**

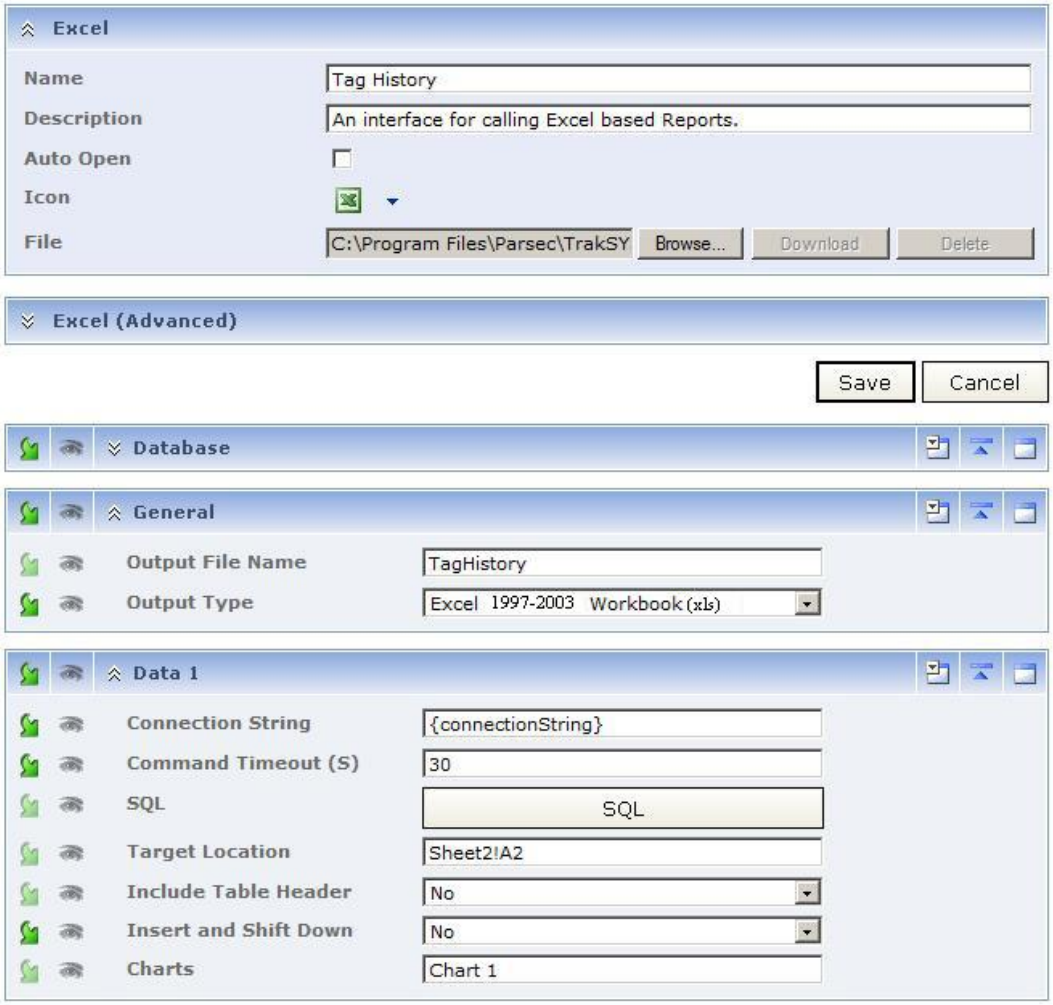
SQL	`SELECT RecordedDateTime, TagValue FROM tTagHistory`
Target Location	Sheet2!A2
Include Table Header	No
Insert and Shift Down	No
Charts	Chart 1

   d. **Filters**

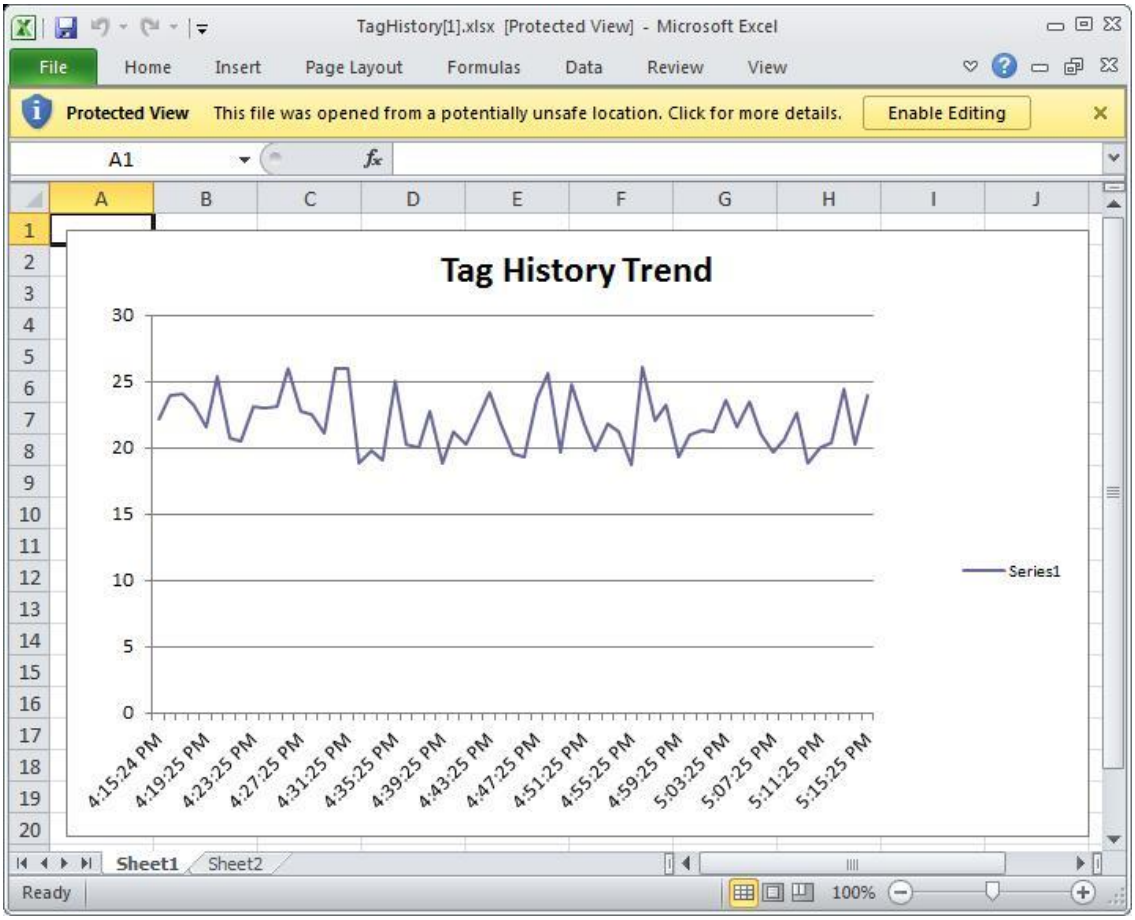
[Section Header]	Click the icon in the section header to hide this section of parameters from the end user. The icon should be grayed-out after clicking it.

   a. **Excel** *[top of page]*

Auto Open	Checked
File	Click the Browse… button, navigate to and select the following file: `C:\Program Files\Parsec\TrakSYS \webTrak\Resources\TagHistoryTemplate.xls`



4. Open the new Tag History report in the WEBTrak tree view to view a trend of history data for a Humidity parameter in an Excel chart.


TagHistory[1].xlsx [Protected View] - Microsoft Excel
File
Home
Insert
Page Layout
Formulas
Data
Review
View
Protected View
This file was opened from a potentially unsafe location. Click for more details.
Enable Editing
A1
Tag History Trend
30
25
20
15
10
5
0
4:15:24 PM
4:19:25 PM
4:23:25 PM
4:27:25 PM
4:31:25 PM
4:35:25 PM
4:39:25 PM
4:43:25 PM
4:47:25 PM
4:51:25 PM
4:55:25 PM
4:59:25 PM
5:03:25 PM
5:07:25 PM
5:11:25 PM
5:15:25 PM
Series1
Sheet1
Sheet2
Ready
100%

Advanced Assignment

There is no Advanced Assignment for this lab.