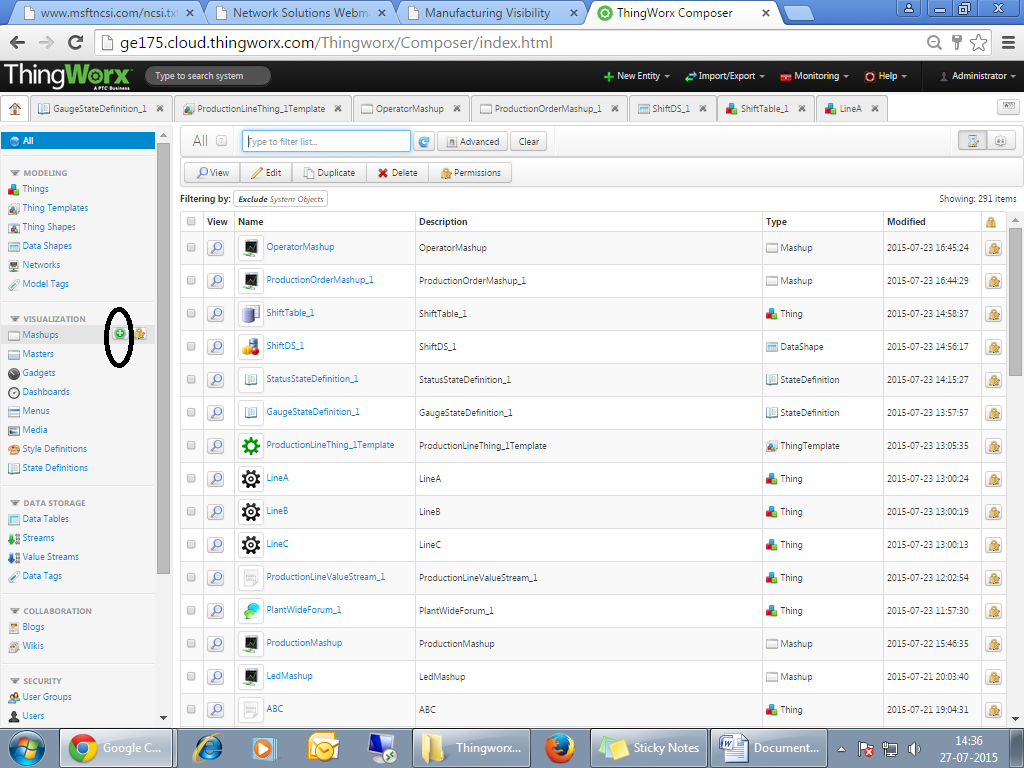
ThingWorx platform to build a simple application using the ThingWorx Composer™ development tool

Step 1:

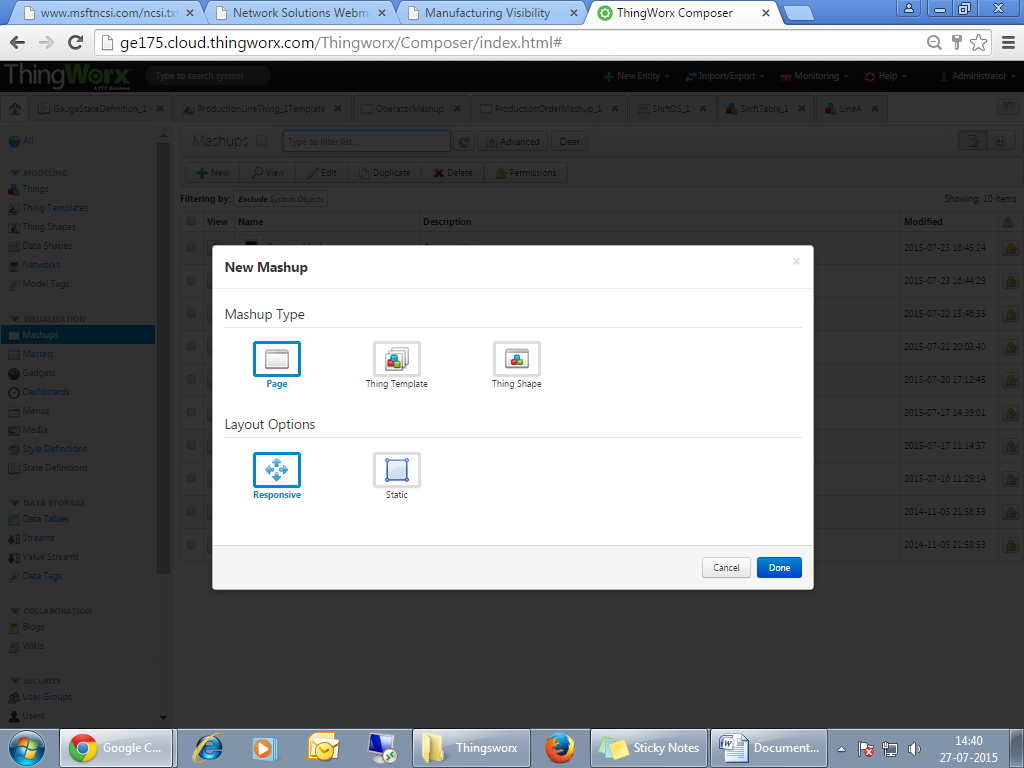
Creating a mashup :



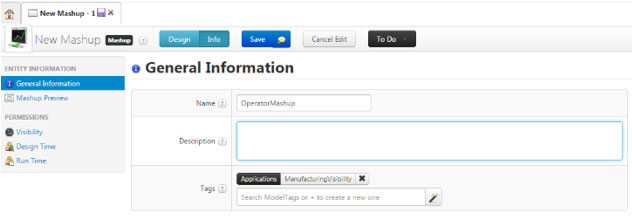
Select the mashup type and Layout Option:

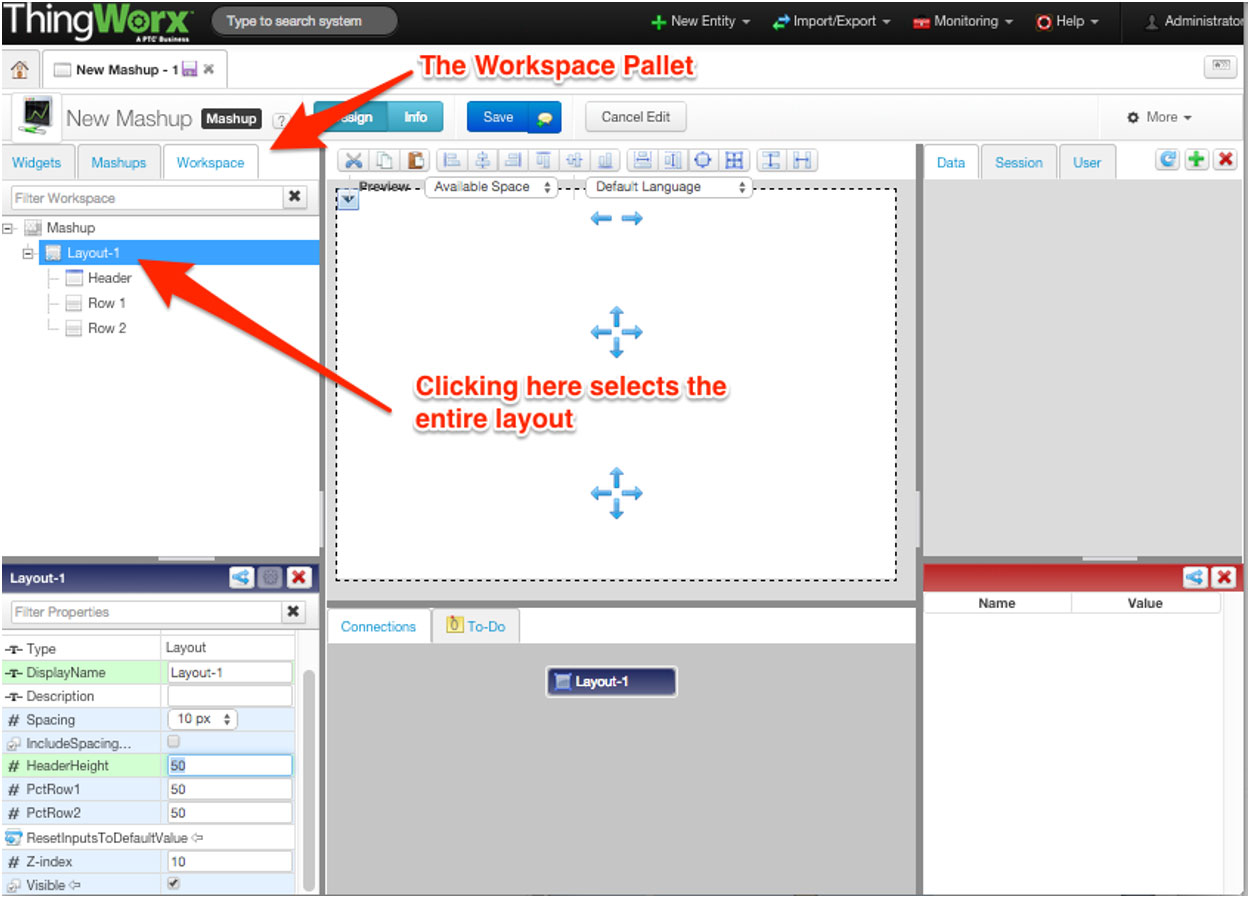
1. mashup type- **page**

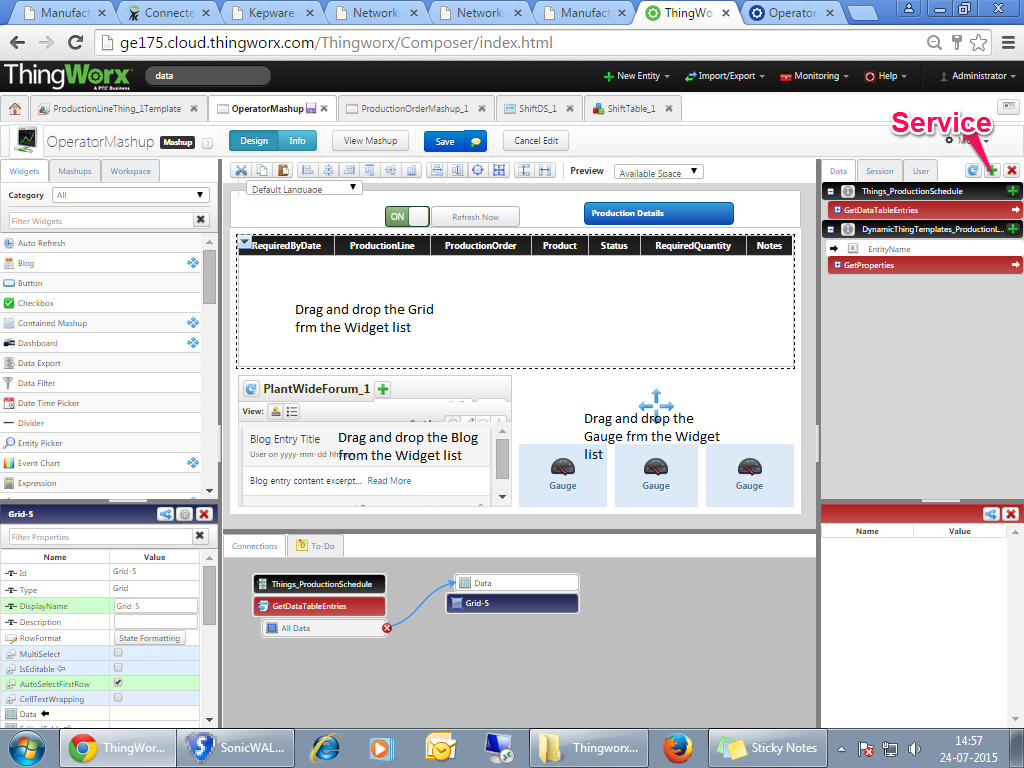
2.layout option-**Responsive**



* Select the Info button
* *Fill out* the following fields.
  + **Name**: OperatorMashup
  + **Tag**: ManufacturingVisibility

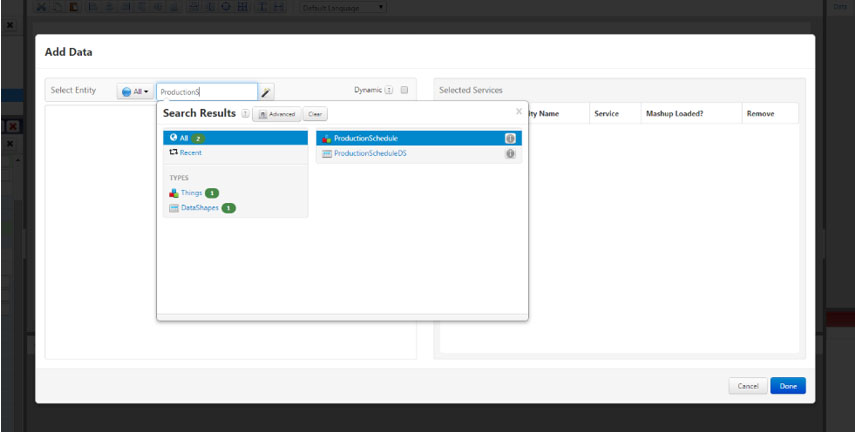


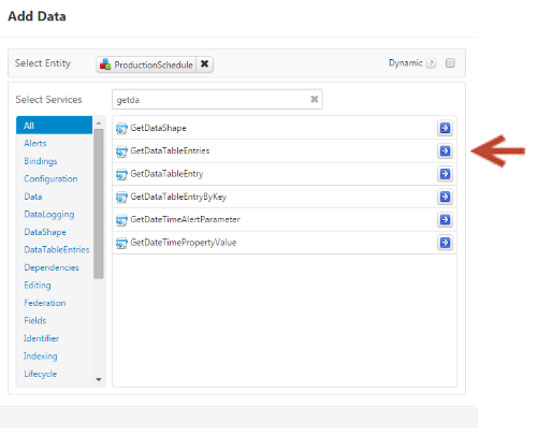
The Page appears like below

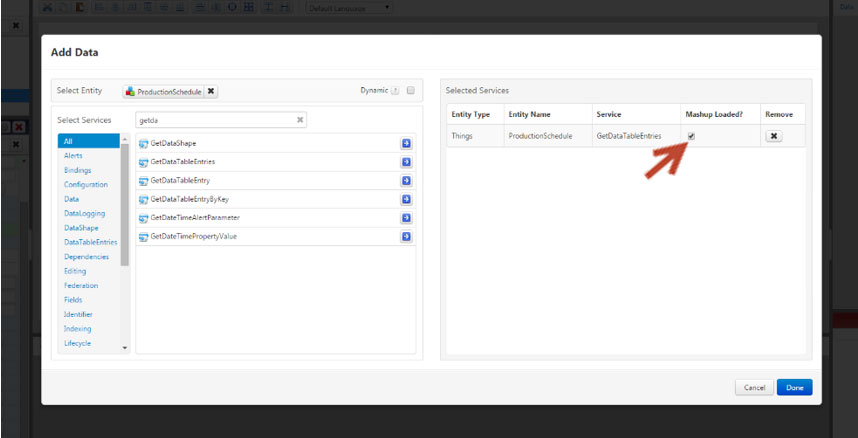
Now , We have to add Widgets,

Adding Service :

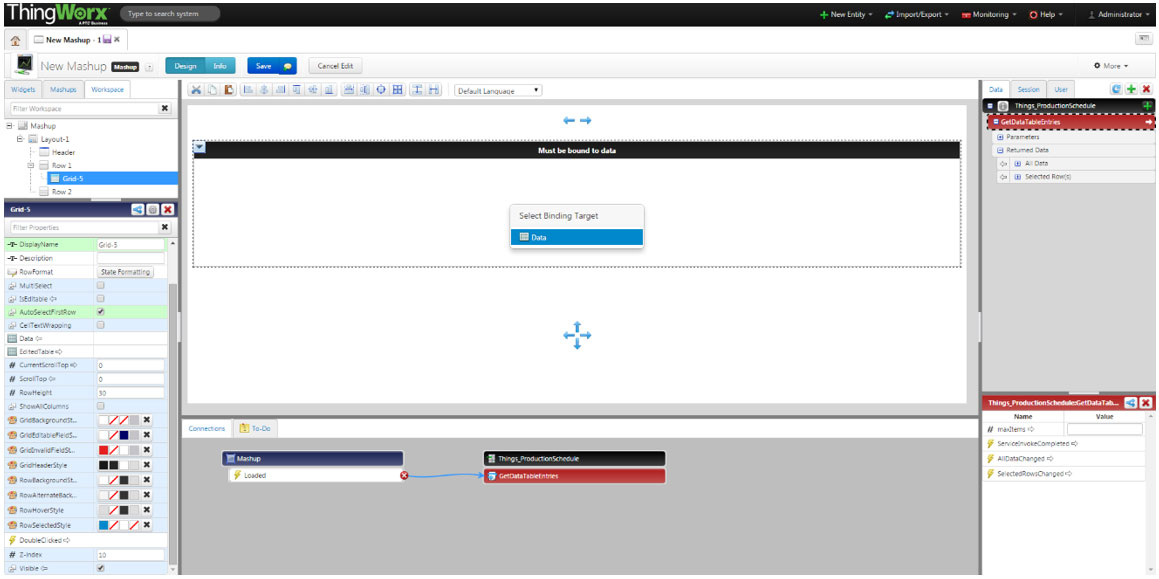
* Click on the green Symbol as show above,
* Select the Entity name: **ProductionSchedule.**
* Add the **GetDataTableEntries** service by clicking on the blue arrow next to it. It should show up under **Selected Services**

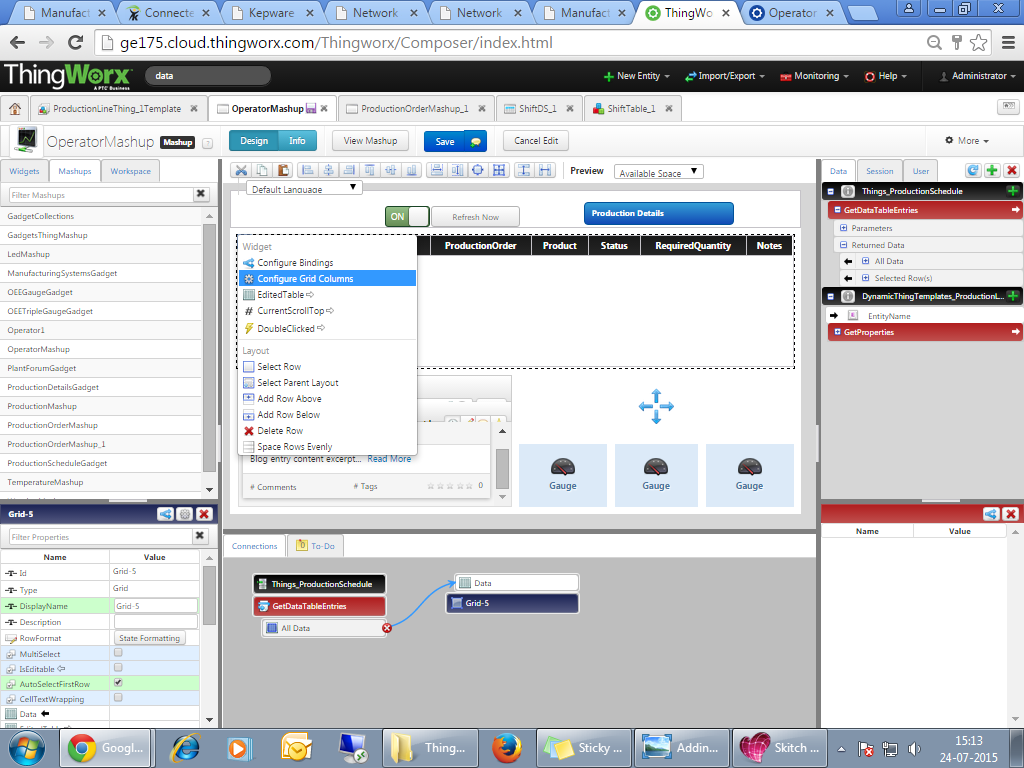


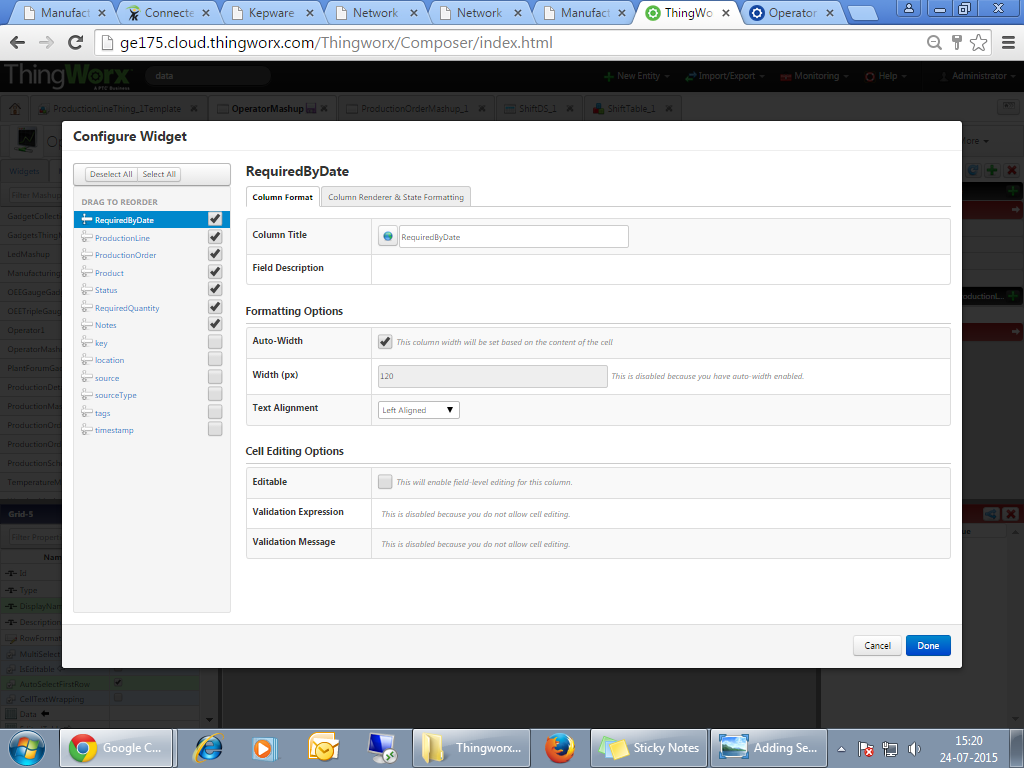


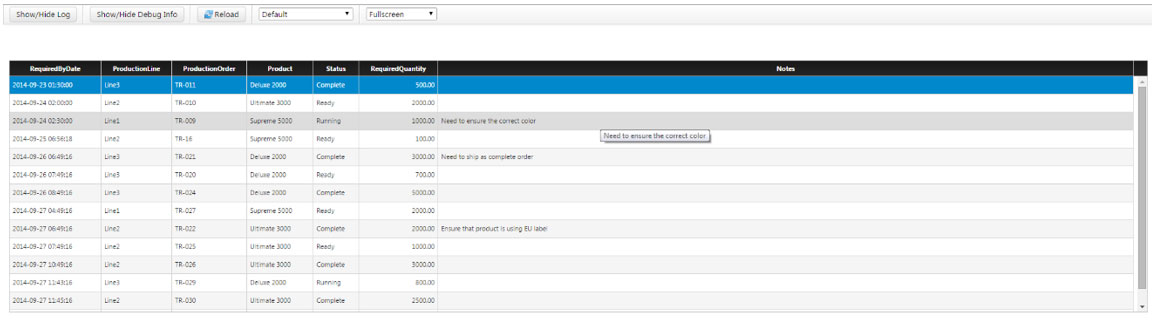


Binding Data:To *Bind* **All Data** from **Things\_ProductionSchedule:GetDataTableEntries** to the grid *drag* **All Data** and *drop*on the grid. A popup with possible binding locations will show up. *Click* on **Data**.



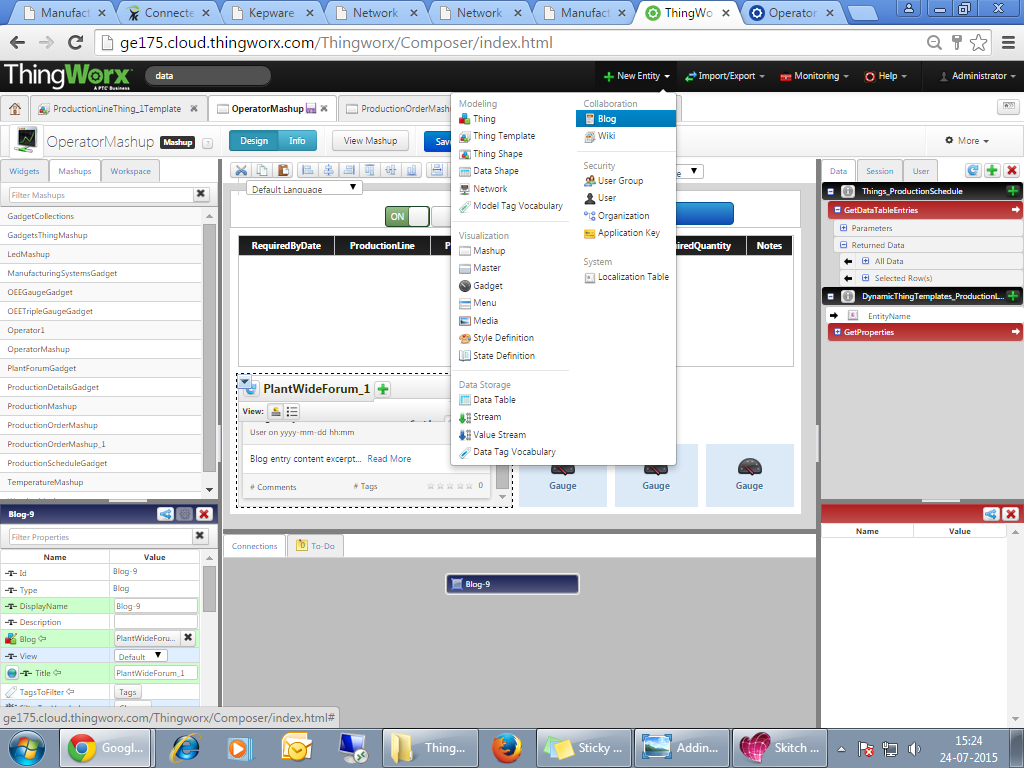


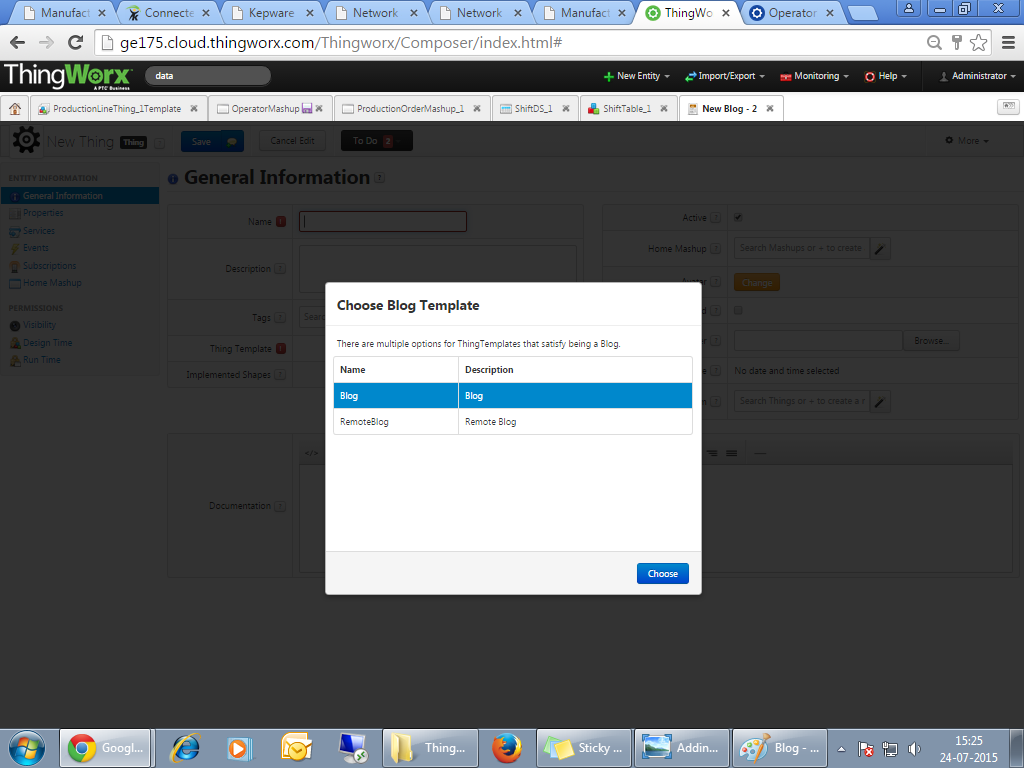




Step 2:

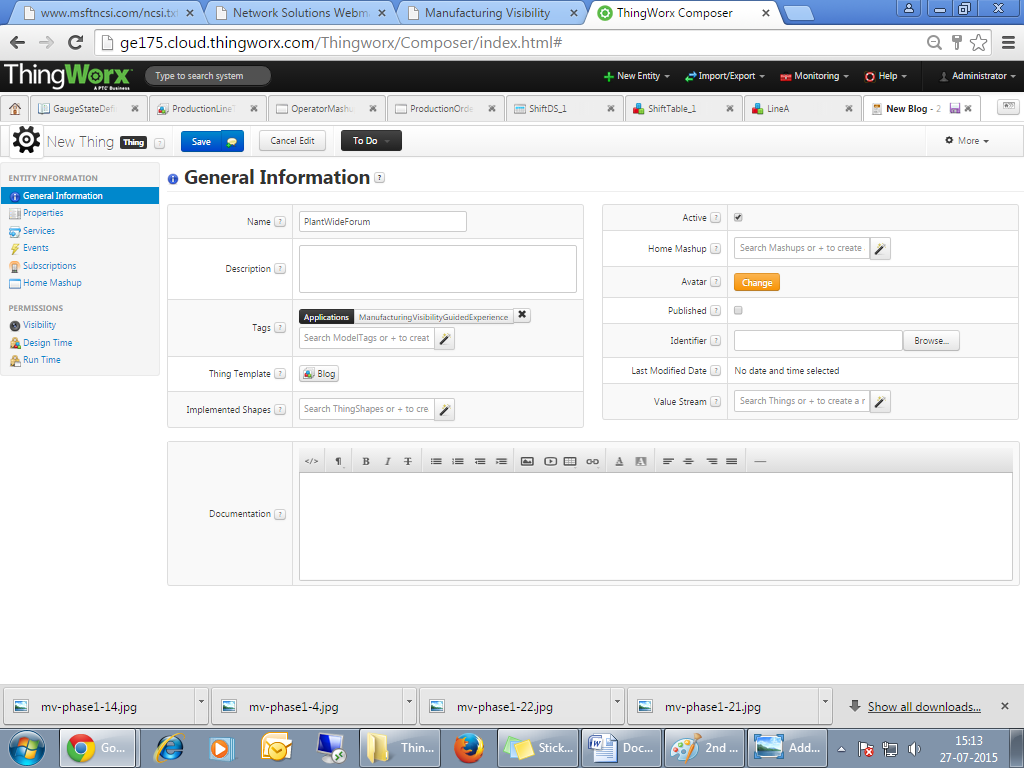
Creating Blog



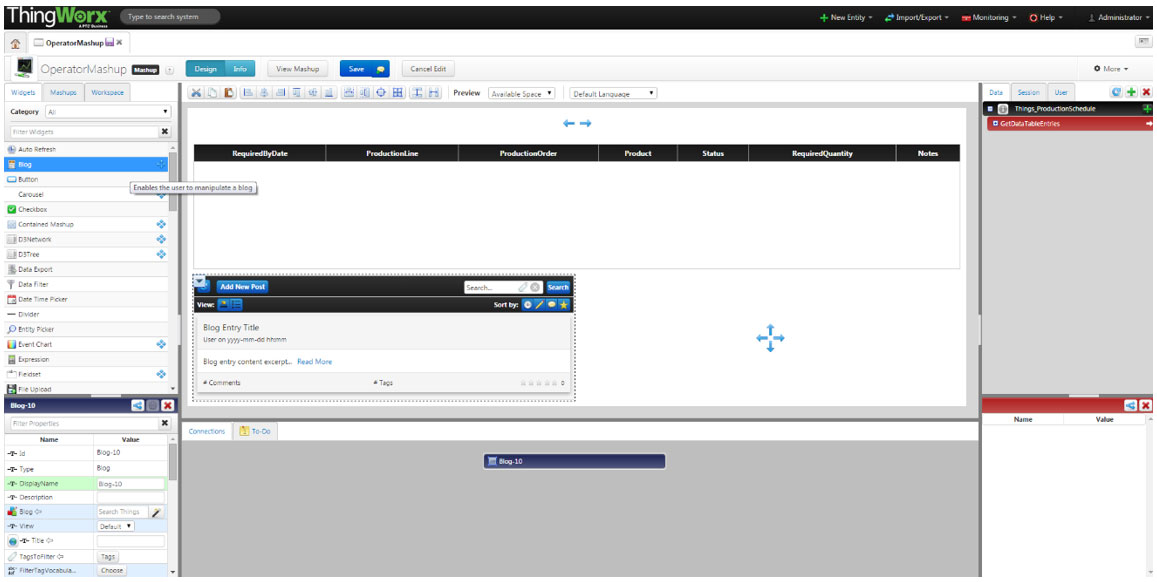


Update the following information:

* Update it with the following information:
  + **Name**: PlantWideForum
  + **Tag**: ManufacturingVisibility
  + Click **Save**.

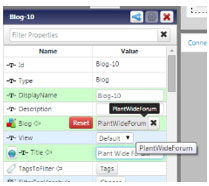


* Go back to the **OperatorMashup**.Add a horizontal layout widget to the second (lower) row of **OperatorMashup**.
* Drag and Drop the Blog Widget from the Widget list

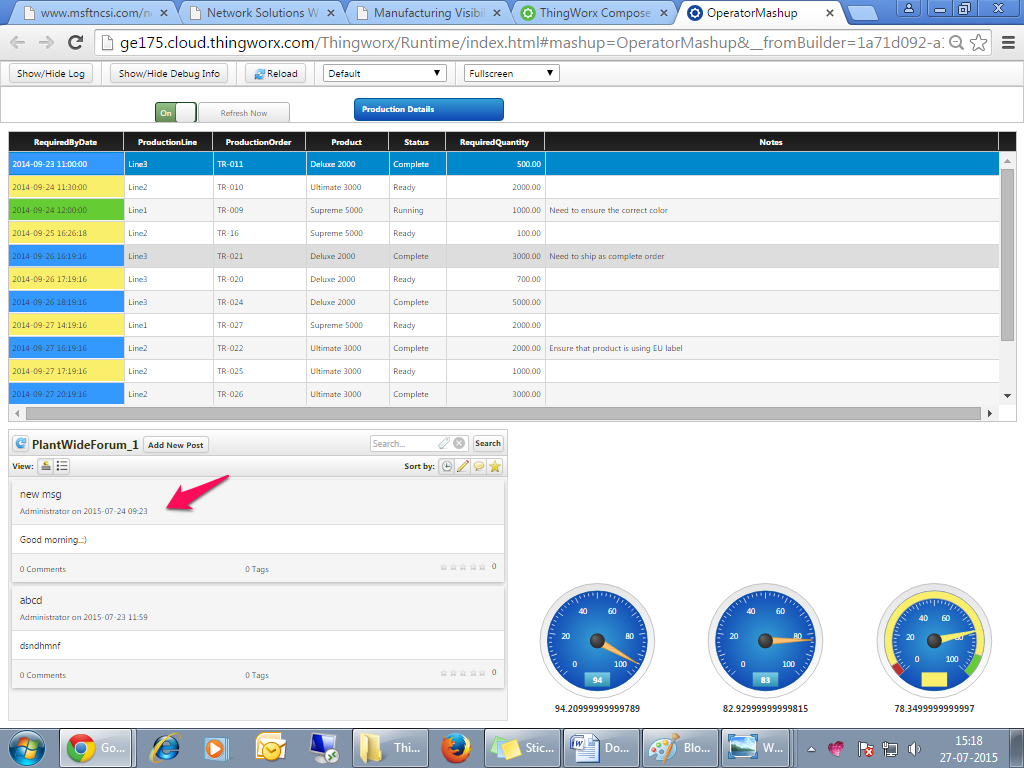


Set the following properties on the Blog Widget

* **Title**: “Plant-Wide Forum”
* **Blog**: PlantWideForum



* Then click on view mashup.
* Add the following information:
  + **Title**: “Notice”
  + **Body**: “ThingWorx is being added as a manufacturing tool. “
  + *Click* **Post**.

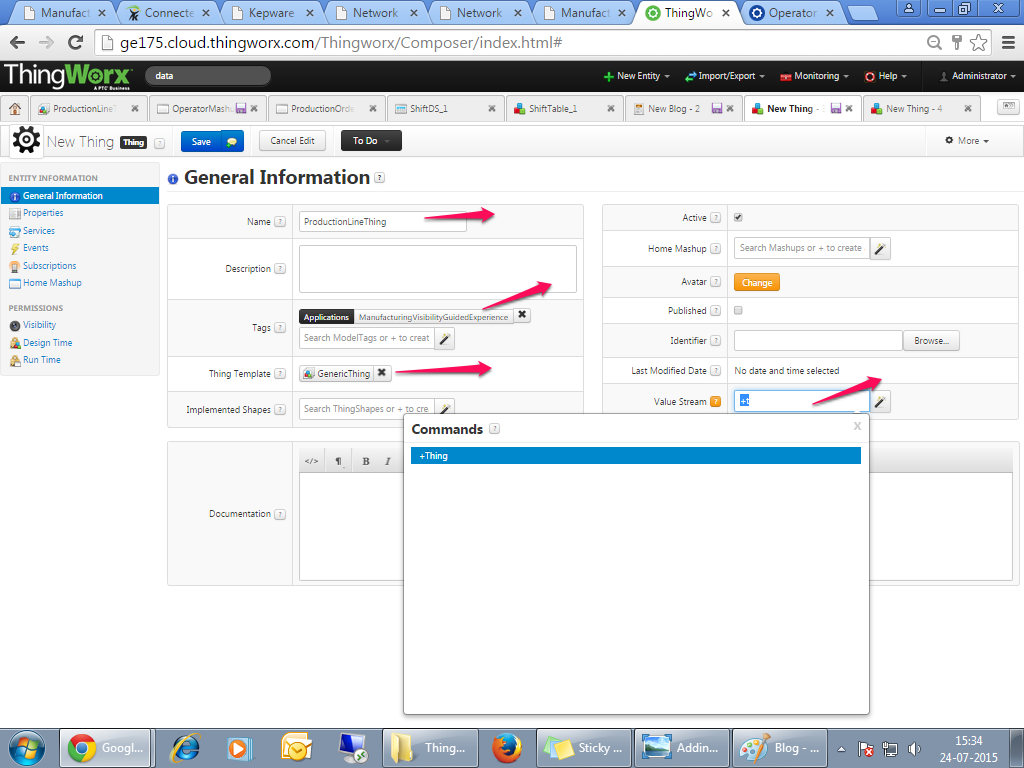


Step 3:Creating Things

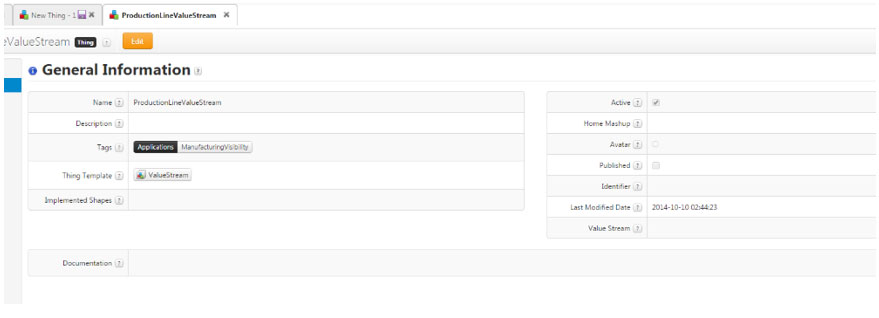
Create a Thing for the Production Line by clicking the **green plus icon** on the Home menu next to Things

* **Name**: “ProductionLineThing”
* **Tags**: ManufacturingVisibilityThing
* **Template**: GenericThing



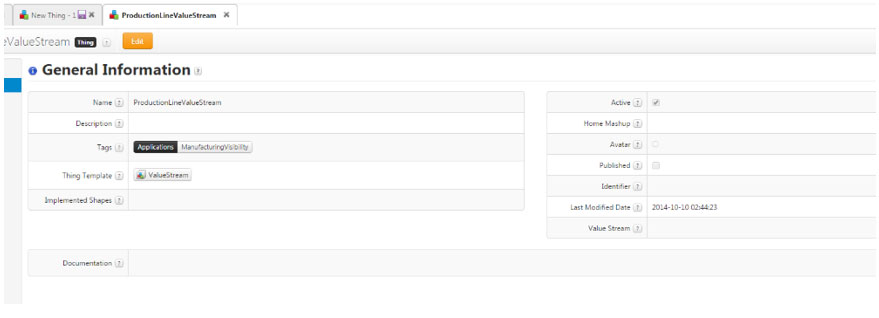


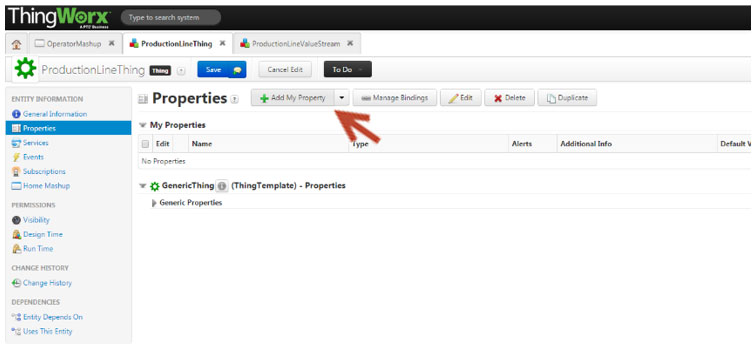
* *Create* a **ValueStream** for this Thing.
  + Type “+” in the Value Stream field.
  + Accept the +Thing command displayed



Enter these values to describe your new **ValueStream**

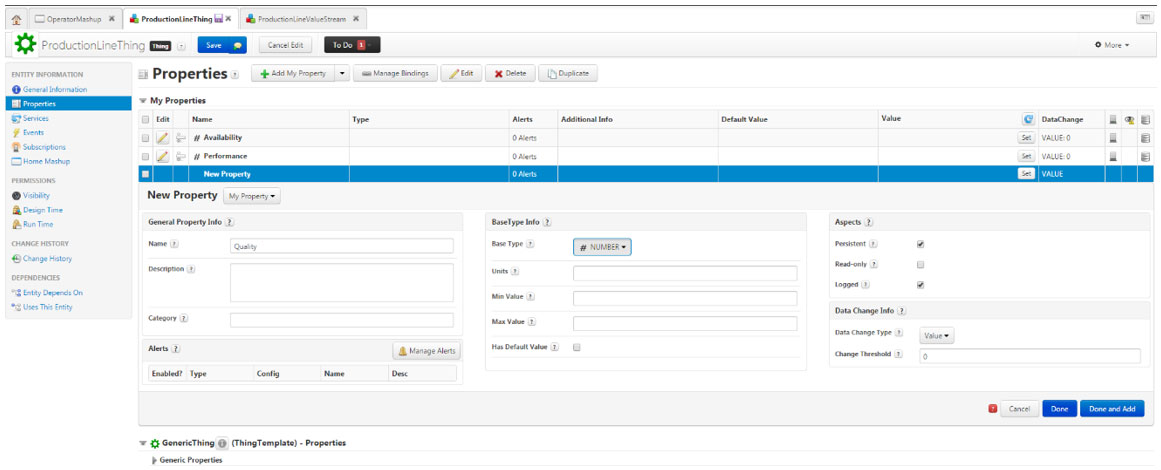
* + **Name**: “ProductionLineValueStream”
  + **Tag**: ManufacturingVisibility
  + **Thing Template**: ValueStream
  + *Save* the **ProductionLineValueStream** then save the **ProductionLineThing**.

Add properties to **ProductionLineThing** by clicking on **Properties** underneath Entity Information, then clicking **Add My Property** in the Properties window. Add each of the Properties listed below



Availability

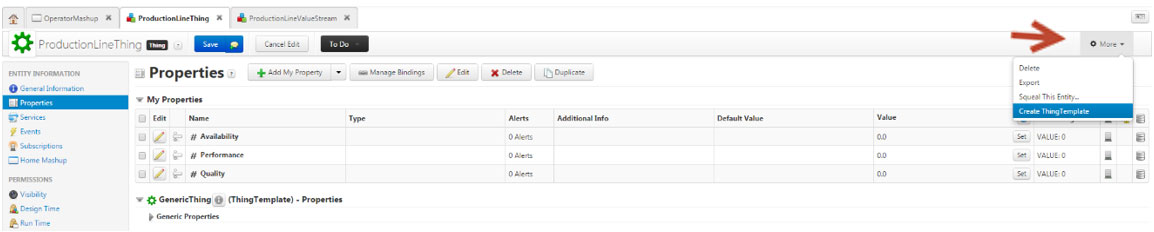
* **Name**: “Availability”
* **Base Type**: Number
* **Persistent**: Yes
* **Logged**: Yes
* *Click* **Done** and **Add**.



* Same way add **Performance**, and **Quality** along with same base type, persistent, Logged and then Save.

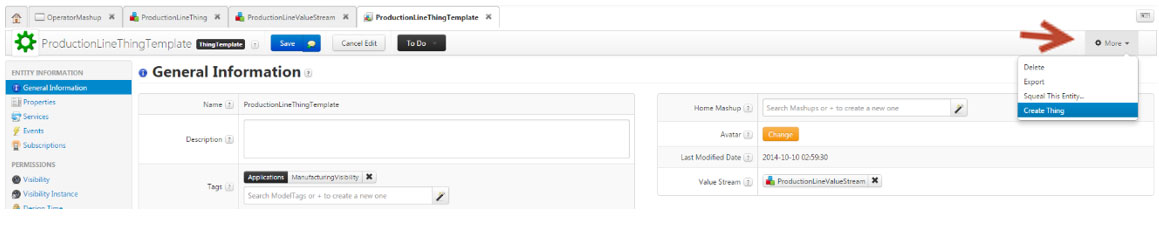
Create a template from **ProductionLineThing** by clicking on the **More** button it the top right corner

* **Name**: “ProductionLineTemplate”
* **Tag**: ManufacturingVisibility
* **Value Stream**: ProductionLineValueStream

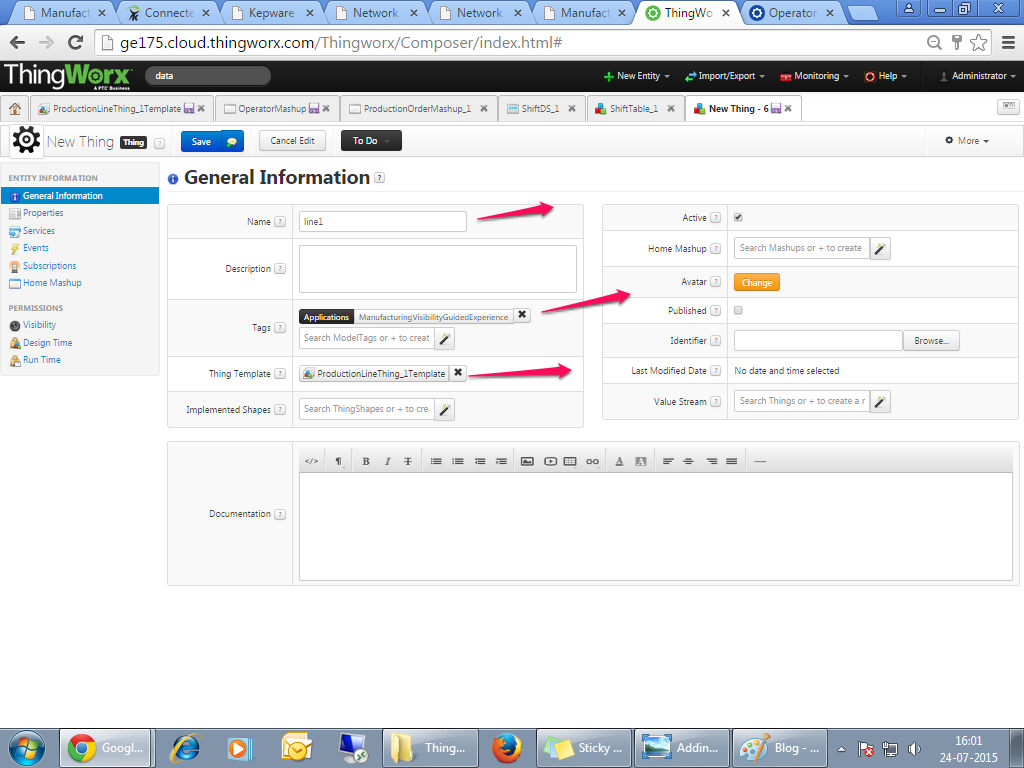


* Click “Save” to save your ProductionLineTemplate
* *Delete* **ProductionLineThing**. To do this, navigate to **Things** in home tab, check the checkbox next to the ProductionLineThing and then click the **Delete** button.

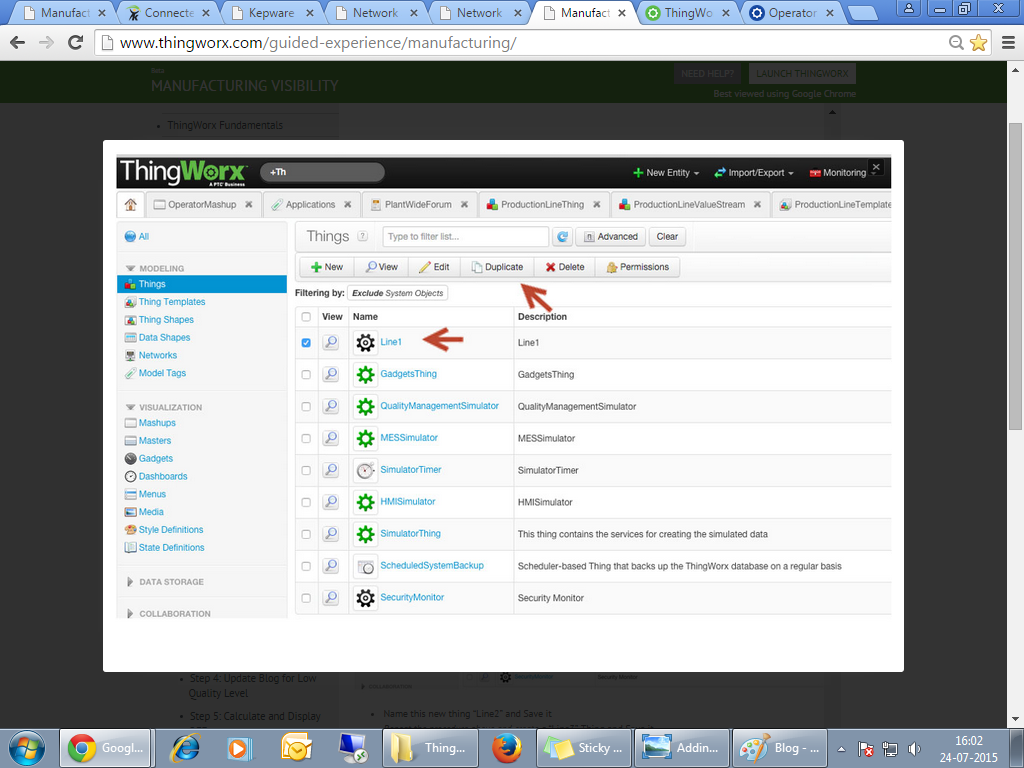
Create a Thing based on **ProductionLineTemplate** via the **Create Thing** option from the **More** menu on the**ProductionLineTemplate**.



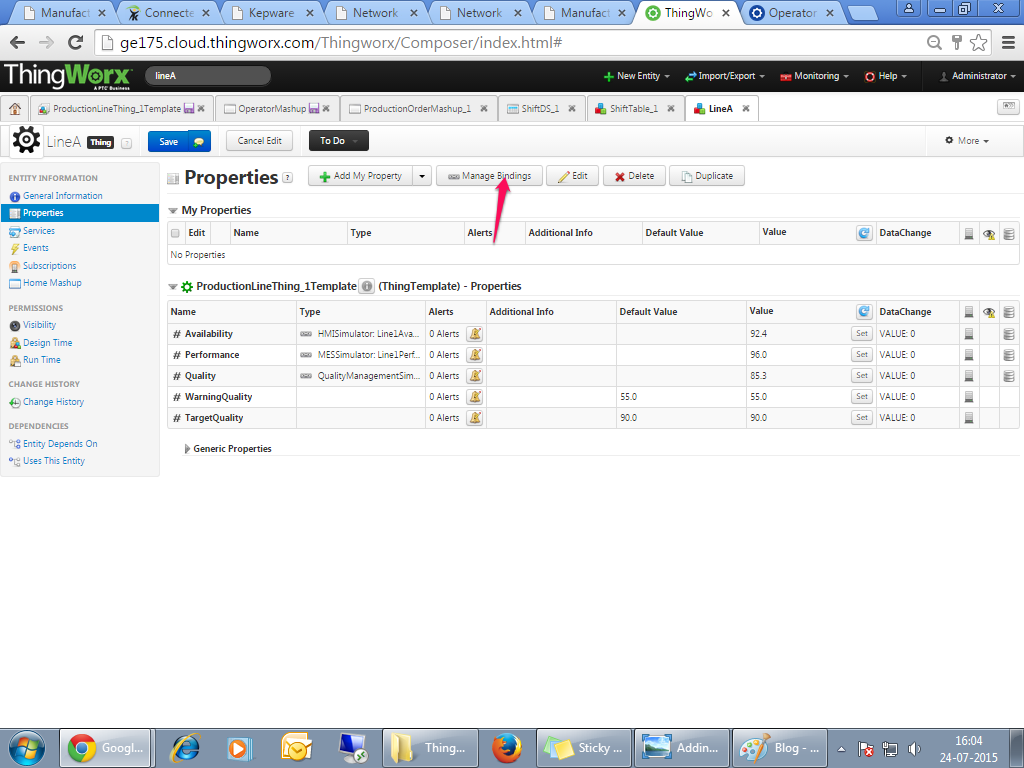
* **Name**: “Line1”
* **Tag**: ManufacturingVisibility
* **Thing Template**: ProductionLineTemplate (Already Chosen)
* Save Line1



* *Create* a “Line2” and “Line3” Thing by duplicating the Line1 Thing
* *Navigate* to the “*Things*” section under Modelling.
* *Check the column to the left of Line1 and select the “Duplicate” button*

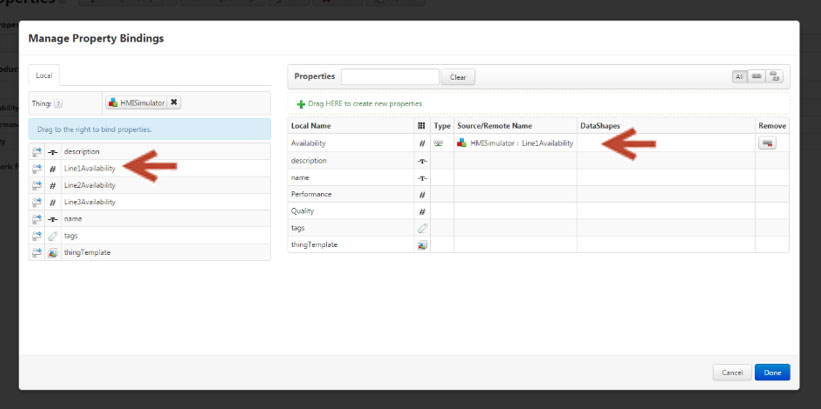


* Name this new thing “Line2” and " Line 3" , Then Save it.
* *Click* on the **Manage Bindings** button.



*Bind* manufacturing simulator data to “Line1” Thing

* For **Thing** *choose* **HMISimulator** then *bind* Availability property to **HMISimulator:Line1Availiblity**by *dragging* and *dropping* Line1Availability from the left-hand side over onto the Availability property on the right-hand side.

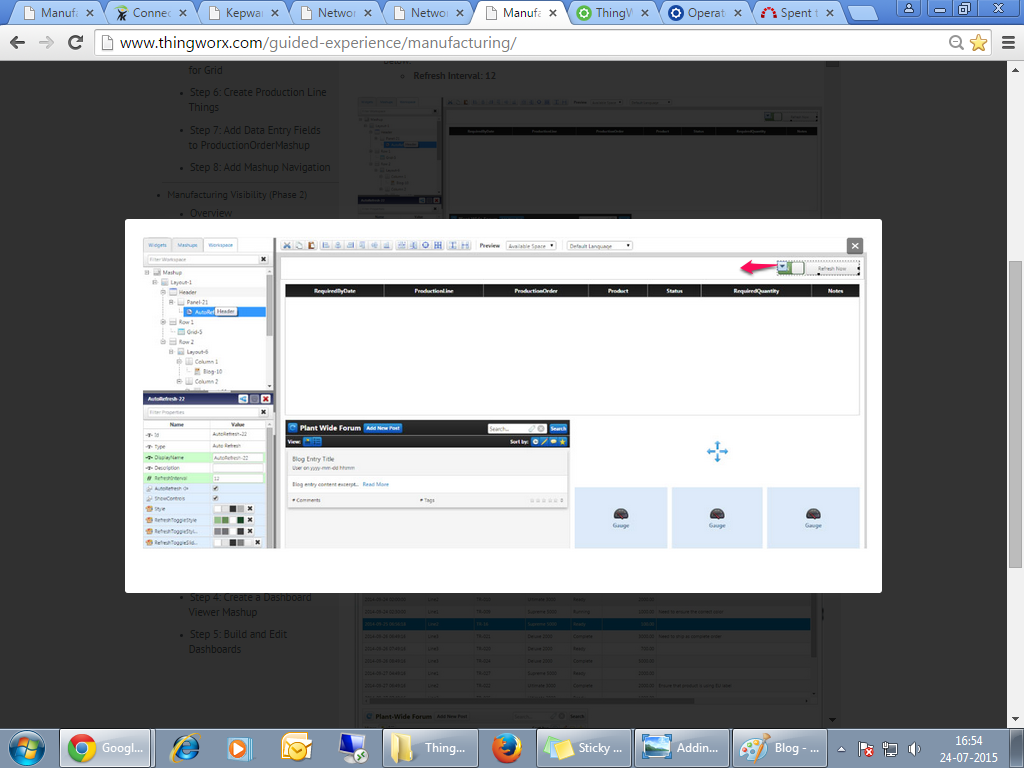


* Next and choose **MESSimulator**. Bind the **Performance** property to**MESSimulator:Line1Performance** in the same way as before.
* *Then choose* **QualityManagementSimulator** then *drag and drop* **Quality** Property to**QualityManagementSimulator:Line1Quality**.
* *Repeat* the same steps for **Line2** Thing and Line3 Thing.
* **Line2** Thing
  + **Availability** Property binds to **HMISimulator:Line2Availiblity**
  + **Performance** Property binds to **MESSimulator:Line2Performance**
  + **Quality** Property binds to **QualityManagementSimulator:Line2Quality**
* **Line3** Thing
  + **Availability** Property *binds* to **HMISimulator:Line3Availiblity**
  + **Performance** Property binds to **MESSimulator:Line3Performance**
  + **Quality** Property binds to **QualityManagementSimulator:Line3Quality**
* *Don’t forget to Click* **Done** and **Save** after each one. we need to add gauge's, to represent these data.

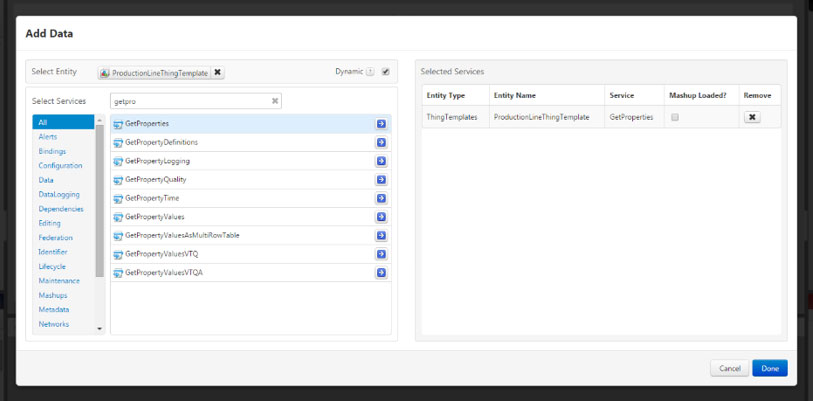
## Step 4:Display Production Line Data on Mashup

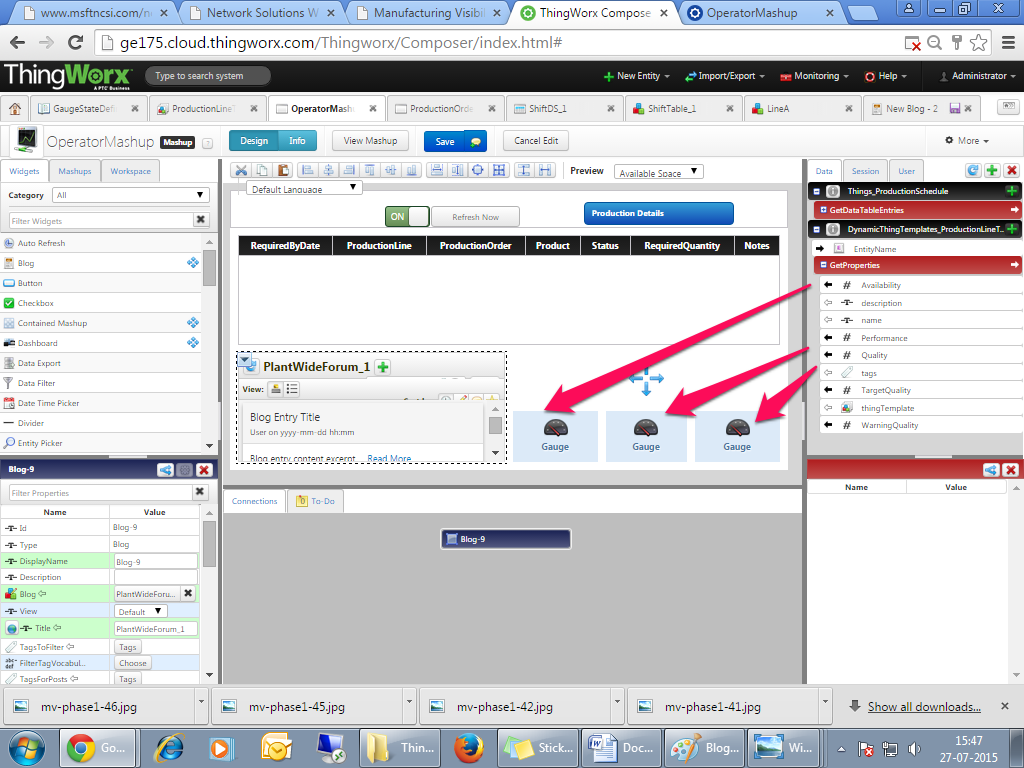
## C:\Users\Administrator\Desktop\Thingworx_picx\mv-phase1-42.jpg

* *Edit* **OperatorMashup**.
* *Add* a vertical layout widget to the lower right-hand column.
  + **Header**: None
  + **Rows**: 2
* Select “Done”
* *Add* a horizontal layout widget to the bottom row of the previously added layout widget.
  + **Sidebars**: None
  + **Columns**: 3
* Select “Done”
* *Add* a **Gauge** widget to each of the three newly created columns.



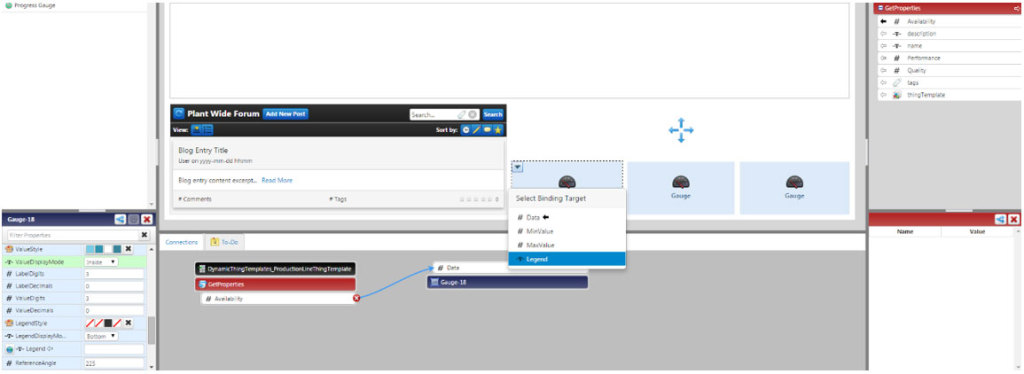
* *Add* a Service to the Mashup. *Check* the **Dynamic** checkbox in order to find the Service listed below.
  + Entity Name: ProductionLineTemplate
  + **Service**: GetProperties
  + **Dynamic**: Yes
  + **Mashup Loaded**? : Yes

*Expand* the newly added service. *Add* **Availability** from**DynamicThingTemplates\_ProductionLineTemplate:GetProperties** to the Gauge on the far left by*dragging and dropping* it over to the widget and binding it to the **Data** property when prompted.

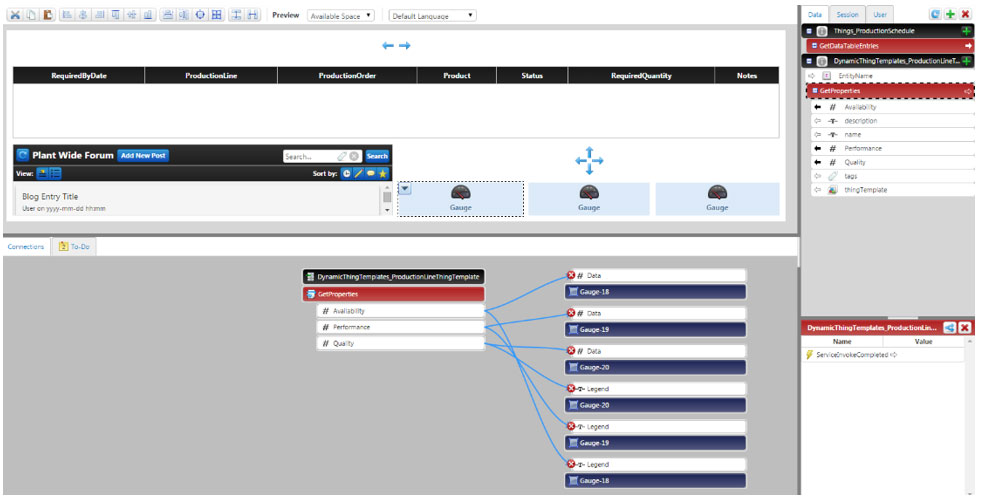


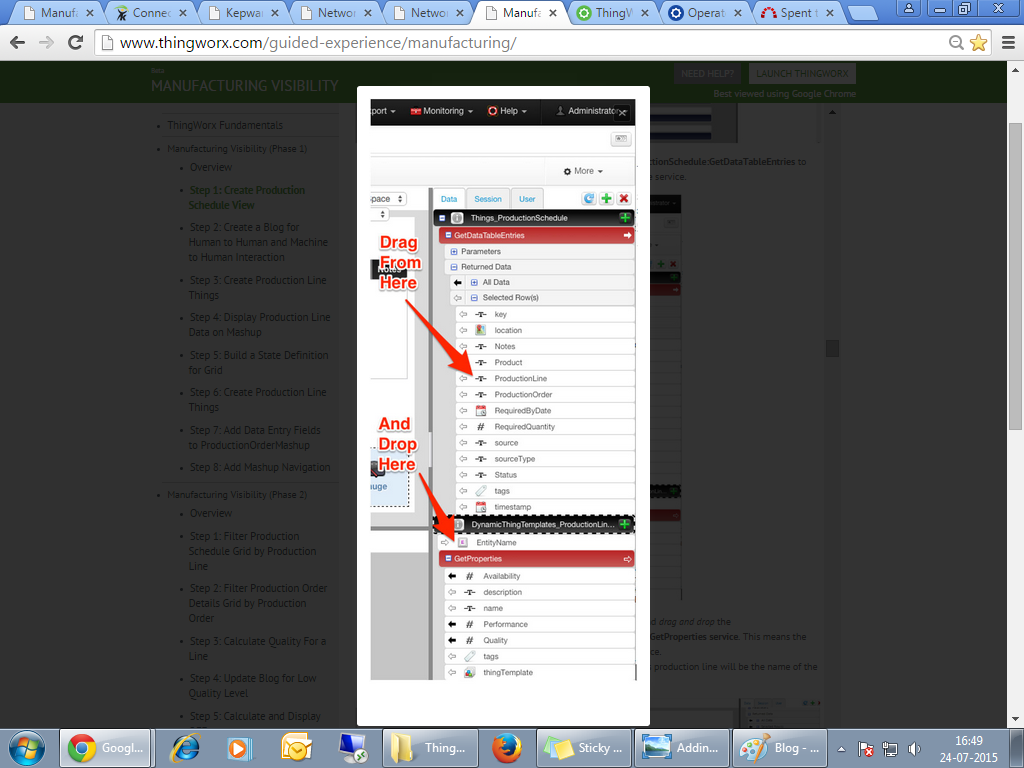
**Legend**: **Availability**

**ValueDisplayMode:** **Inside**

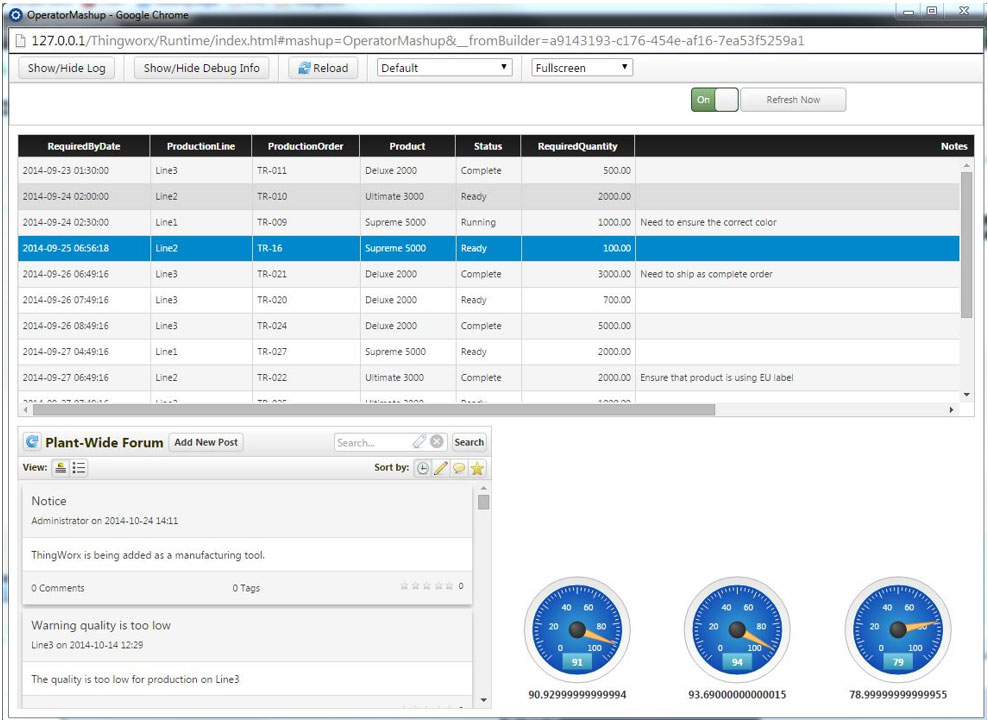
****

* *Drag and drop* **Performance** from **ProductionLineTemplate:GetProperties** to the gauge in the middle and*bind* it to **Data** when prompted.
* *Change* the settings of the middle Gauge.
  + **Legend**: **Performance** (*drag and drop* **Performance** from **ProductionLineTemplate:GetProperties**onto the Gauge widget and *bind* it to **Legend** when prompted)
  + **ValueDisplayMode**: **Inside**
* *Drag and drop* **Quality** from **ProductionLineTemplate:GetProperties** to gauge on the far right and *bind* it to**Data** when prompted.
* *Change* the settings of the right Gauge.
  + **Legend** : **Quality** (*drag and drop* **Quality** from **ProductionLineTemplate:GetProperties** onto the Gauge widget and *bind* it to **Legend** when prompted)
  + **ValueDisplayMode:** **Inside**

*Bind* **ProdutionLine** from the **Selected Row** section of the **ProductionSchedule:GetDataTableEntries** to**EntityName** of **DynamicThingTemplate\_ProductionLineTemplate** service.



* *Add* a **Panel** widget to the header of the top layout widget.
* *Add* **Auto Refresh** widget to far right side of the new header panel and update the widget property shown below.
  + **Refresh Interval: 12**



## Step 5: Build State Definition for Gauge

## *Add* properties to ProductionLineTemplate by *clicking* the Add My Property button from the Properties page of the ThingTemplate.

## C:\Users\Administrator\Desktop\Thingworx_picx\Thimg_19.png

*Add* a WarningQuality property with the below settings. Then *click* **Done and Add.**

* **Name**: “WarningQuality”
* **Base Type**: Number
* **Persistent**: Yes
* **Has Default Value**: Yes
* **Default Value**: 55.

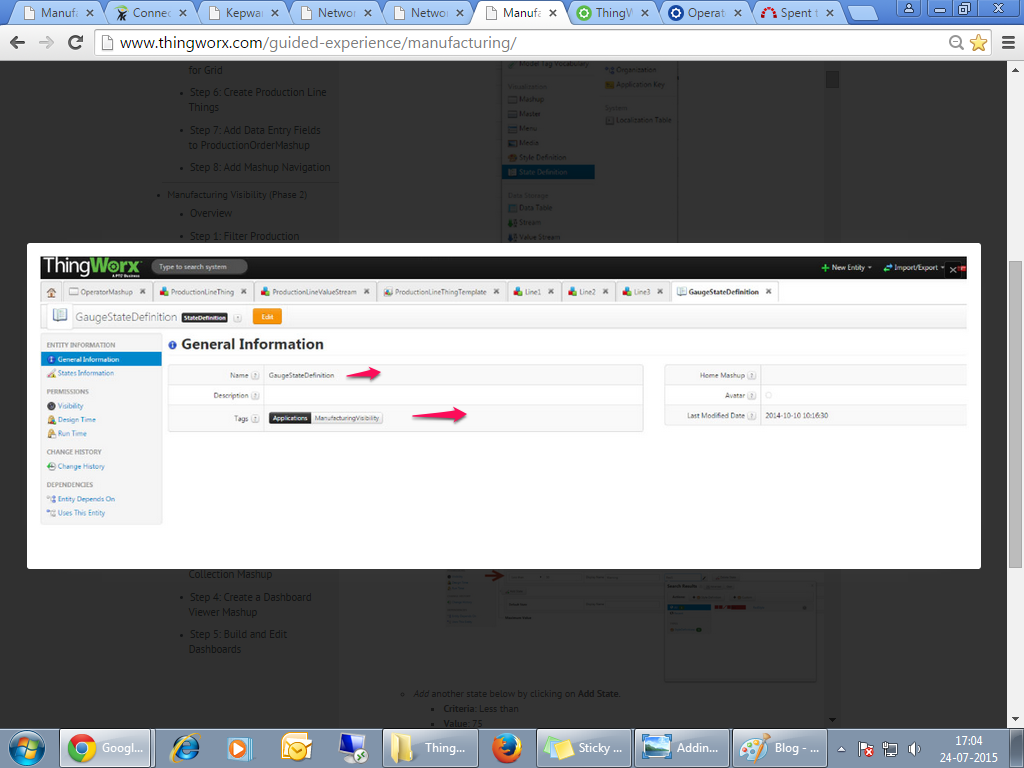
Add a TargetQuality property with the below settings. Then click **Done**. Then, **Save** the Template.

* + **Name**: “TargetQuality“
  + **Base Type**: Number
  + **Persistent**: Yes
  + **Has Default Value**: Yes
  + **Default Value**: 90, Select “Done” and “Save”

Create a new**State Definition** by clicking on the New Entity dropdown at the top and choosing State Definition.

* **Name**: “GaugeStateDefinition”
* **Tag**: ManufacturingVisibility





Click on the **States Information** to configure the states. Set the **States** dropdown to **Numeric Range**. Then, click on **Add State**.

* **Criteria**: Less than
* **Value**: 30
* **Display Name**: “Warning“
* **Style Definition**: RedStyle

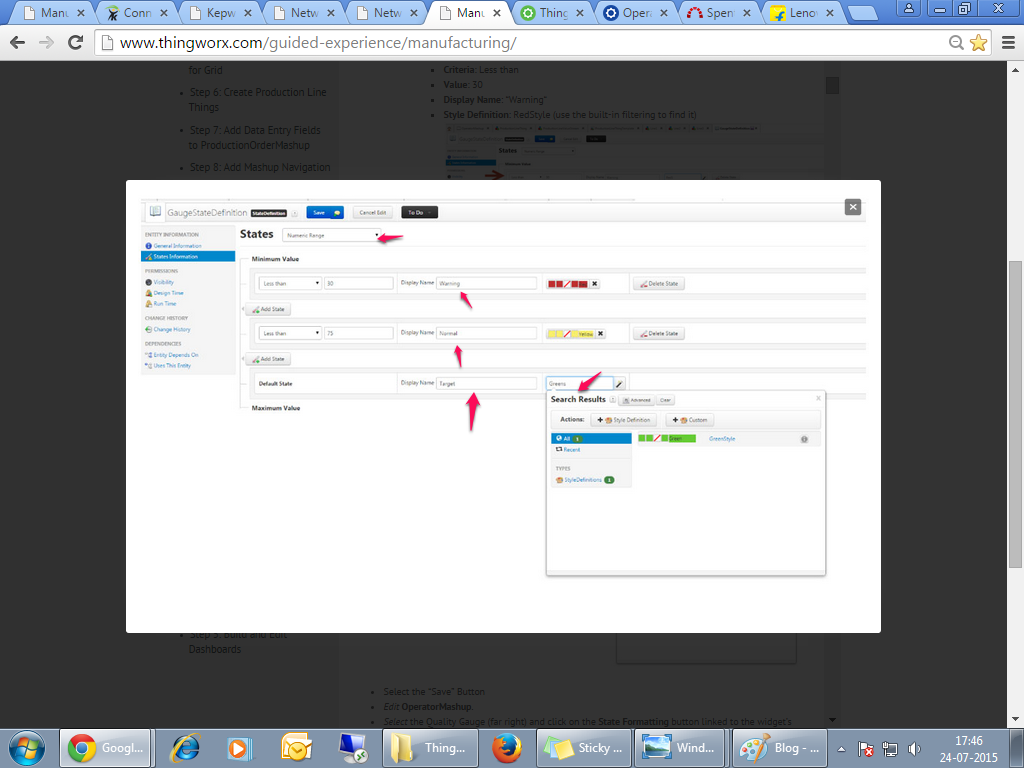
Add another State, below

* + - **Criteria**: Less than
    - **Value**: 75
    - **Display Name**: “Normal“

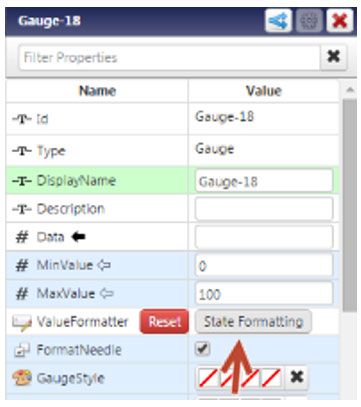
**Style Definition**: YellowStyle

*Setup* **Default State**. *Click* **Save** when done.

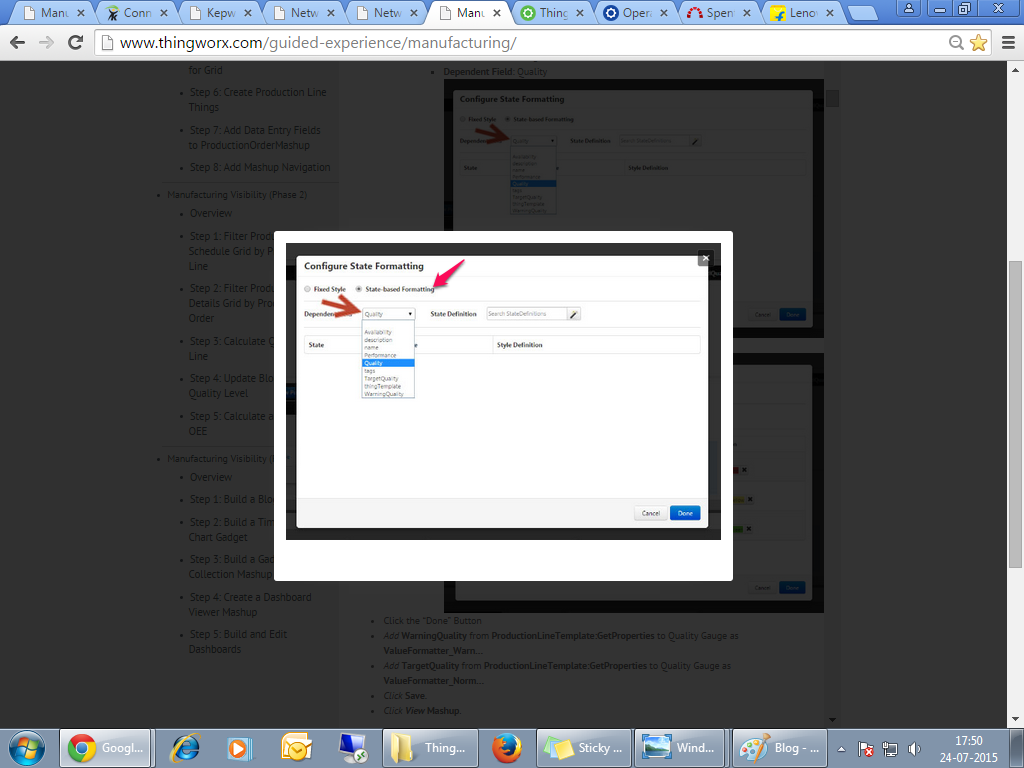
* + - **Display Name**: “Target“
    - **Style Definition**: GreenStyle



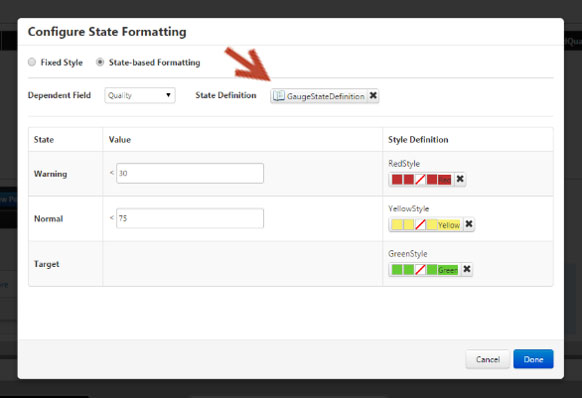
* *Edit* **OperatorMashup**.
* *Select* the Quality Gauge (far right) and click on the **State Formatting** button linked to the widget’s**ValueFormatter** property in the Widget Properties section on the bottom left.

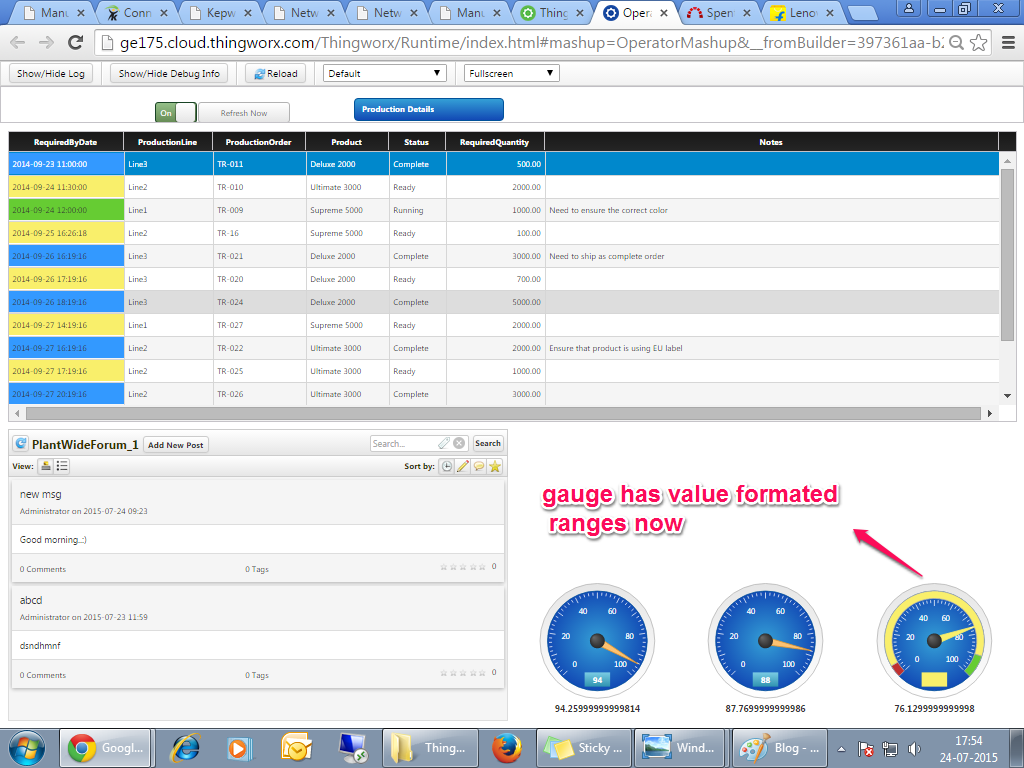


**Dependent Field**: Quality



**State Definition**: GaugeStateDefinition





## Step 6:Build a State Definition for Grid

## C:\Users\Administrator\Desktop\Thingworx_picx\Thimg_20.png

## Name: StatusStateDefinition

## Tag: ManufacturingVisibility

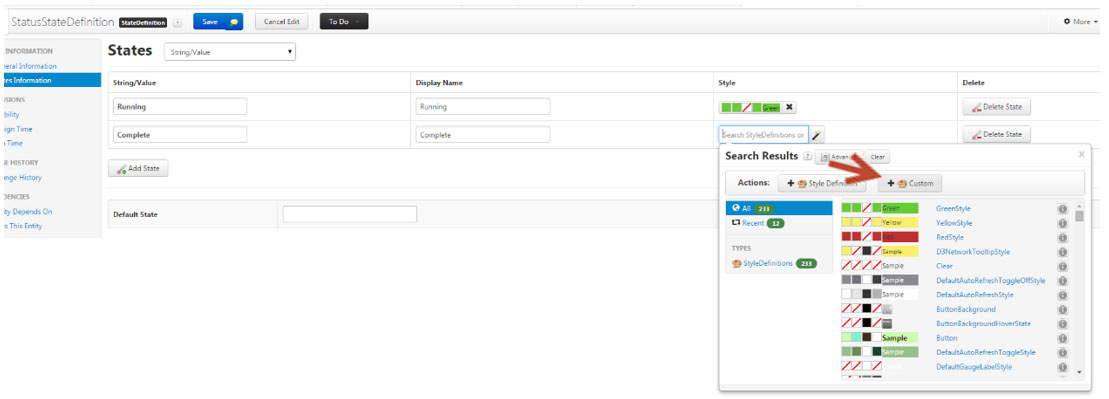
## *Click* on States Information to configure the states. *Set* the States dropdown to String/Value. Then, *click*Add State.

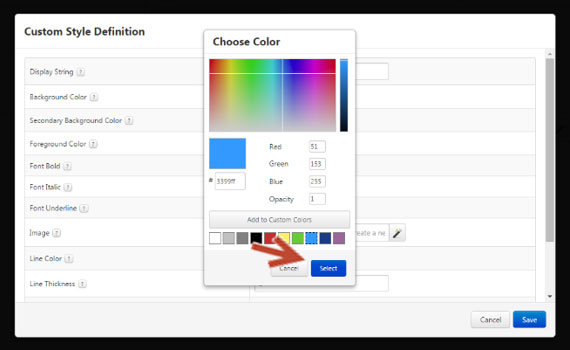
*Add* the first state as described below, and then click **Add State**.

* **String/Value**: Running
* **Display Name**: “Running“
* **Style Definition**: GreenStyle

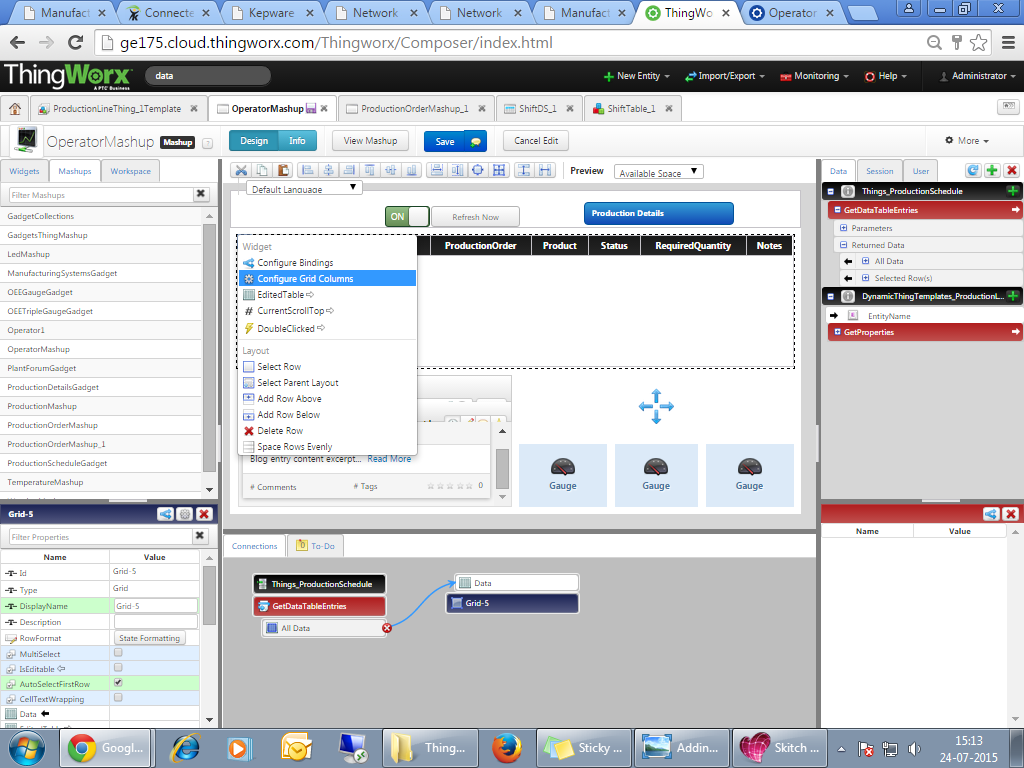
*Add* the 2nd state,

* **String/Value**: Complete
* **Display Name**: “Complete“
* **Style Definition**: create **Custom.**
* **Display String**: Blue
* **Background Colour**: Blue
* **2nd Background Colour**: Blue
* **Line Colour**: Blue.
* *Set* the **Default State** as follows: **Style Definition**: YellowStyle





* *Go* back to the **OperatorMashup**.
* *Select* the **grid** widget.
* *However* the mouse over the top-left corner of the widget to view the widget’s context menu. *Click* on **Configure** **Grid Columns**.

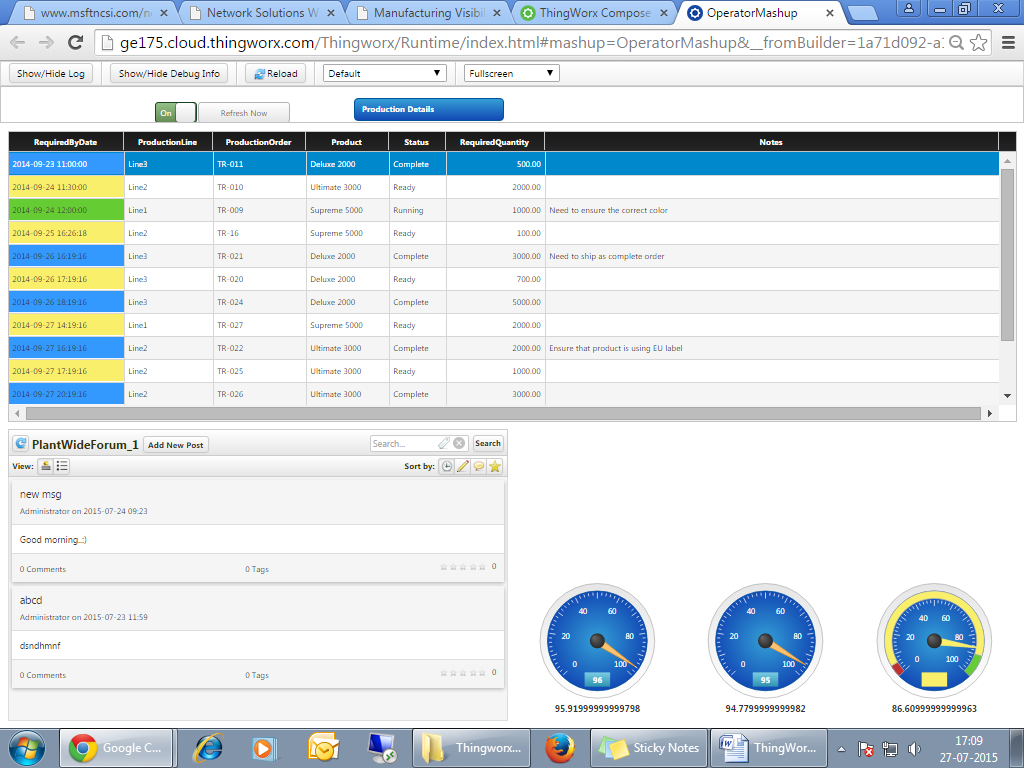




Select the **Status** column on the left.

Click on the **Column Renderer** **& State Formatting** tab.

* Check “**State-based Formatting**”
* **Dependent Field**: Status
* **State Definition**: StatusStateDefinition.



This is a pop up screen, which follows the similar steps,

