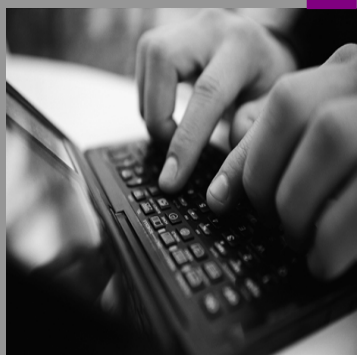


How-to Guide
SAP NetWeaver 2004s

How To... Integrate Visual Composer Applications and BEx Web Templates

Version 1.4 – April 2006

Applicable Releases:
SAP NetWeaver '04s

1 Scenario

Due to the wide variety of tools available in SAP NetWeaver 2004s, integration between applications becomes increasingly important. This paper shows how to build web applications utilizing both Visual Composer and Web Applications built with the Web Application Designer and demonstrates how to make these applications communicate with each other in one coherent framework.

2 Introduction

As of SAP NetWeaver 2004s SPS07, the Adobe Flash runtime within Visual Composer utilizes portal eventing capabilities. Web Applications within the Web Application Designer can raise events to communicate with Visual Composer. In addition, Visual Composer can raise events to communicate with the Web Application Designer. For example, you can build one web dashboard with iViews from Visual Composer and iViews from the Web Application Designer, and pass filter values and navigation states back and forth between these applications. This communication will be transparent to the end user.

The benefits of this communication include being able to modularize and reuse components, both within the Web Application Designer and within Visual Composer. You can build one iView in your portal which has all your runtime variables (dropdown lists that set filters). These will apply globally to all your Visual Composer models, BI Web Applications, .NET Applications, JSP's, BSPs, etc … and you can navigate between multiple applications without losing your runtime variables.

3 Prerequisites

Ensure that your BI System is connected to your Enterprise Portal and has a system alias defined.

Visual Composer must be on SAP NetWeaver 2004s SP7 or higher.

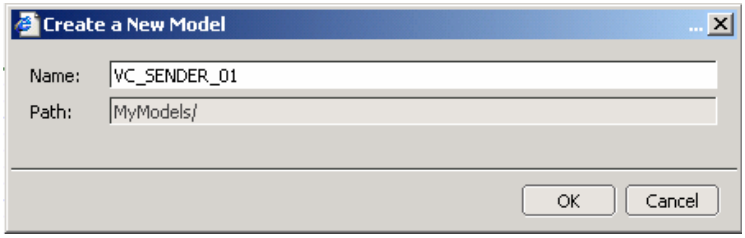
Your client side browser is Internet Explorer 6.0 or above. This solution utilizes ActiveX which isn't supported in browsers outside of Internet Explorer.

4 Step by Step

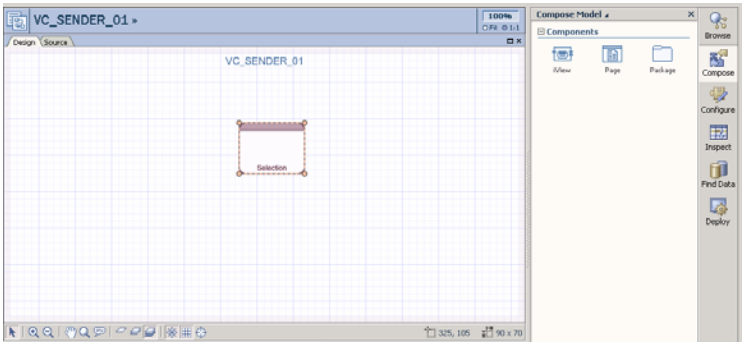
4.1 Sending Events from Visual Composer to Web Application Designer

This scenario will describe how to set filters within Visual Composer using a dropdown list within a form. These dropdown list values will be applied to your Web Application Designer report as well. The example that is created by this exercise demonstrates how to create a dropdown list for Calendar Year within the Web Application Designer. Your Visual Composer application will be filtered by the value selected from the BI Web Application.

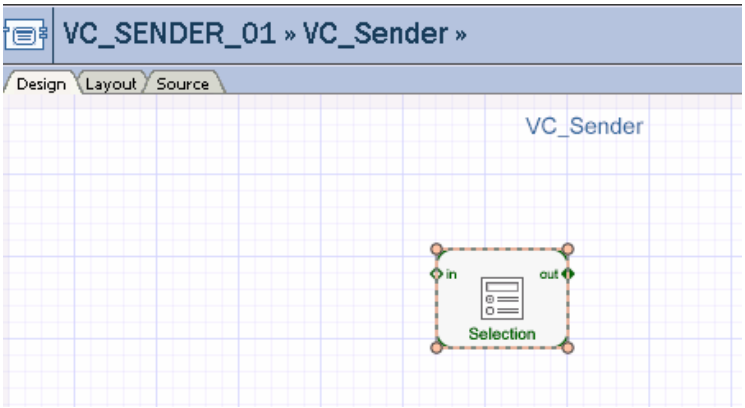
1. Create a New Model within VC.



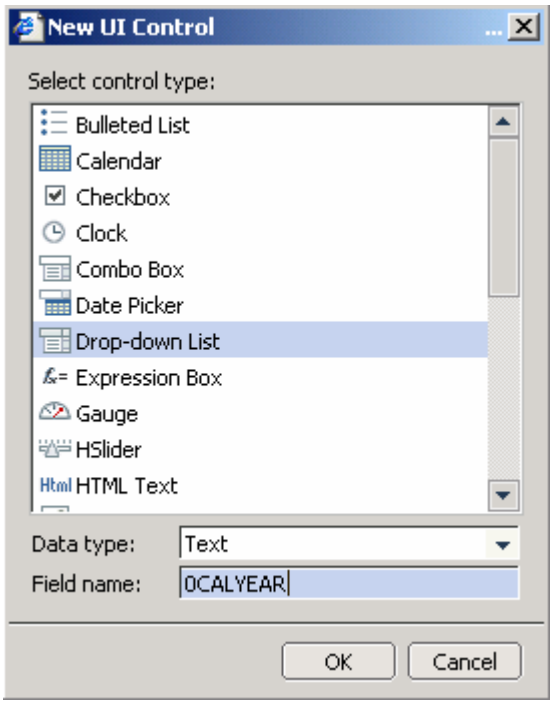
2. Drag and drop an iView within your model and name iView “VC_Sender”.



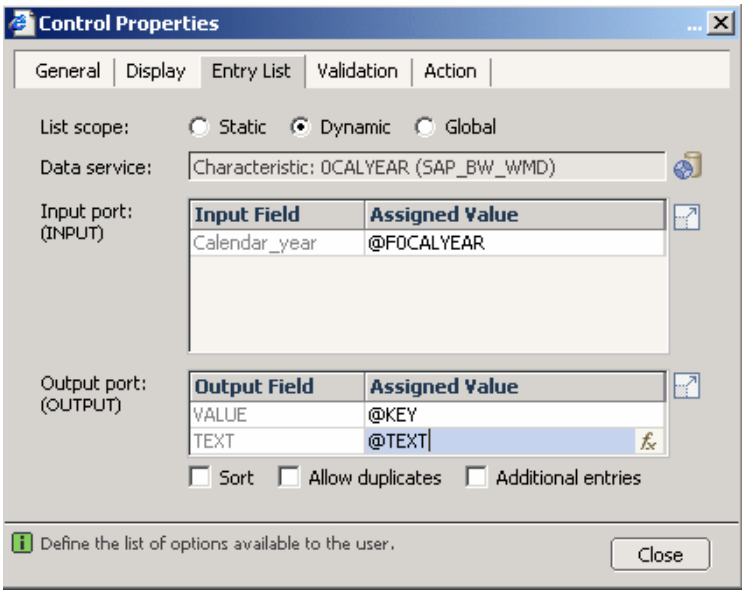
3. Drag and drop a form within your model and name this form “Selection”.



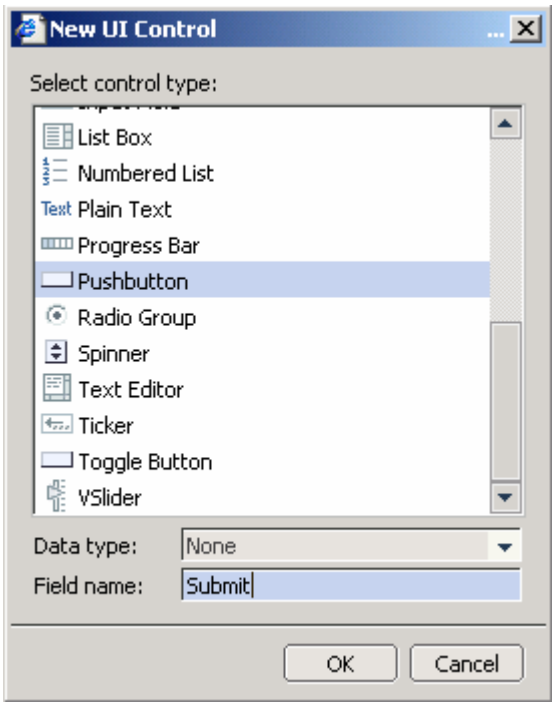
4. Add a dropdown list within your form and name this field “0CALYEAR”. Keep in mind that fields cannot start with “0” within VC and the field will automatically be renamed to “F0CALYEAR”.



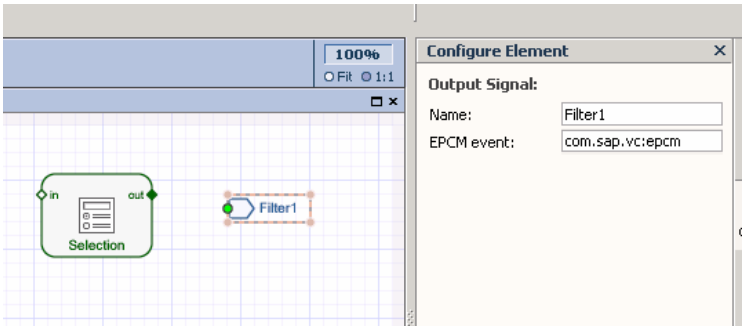
5. Specify either a static or dynamic dropdown list in the properties of this object.



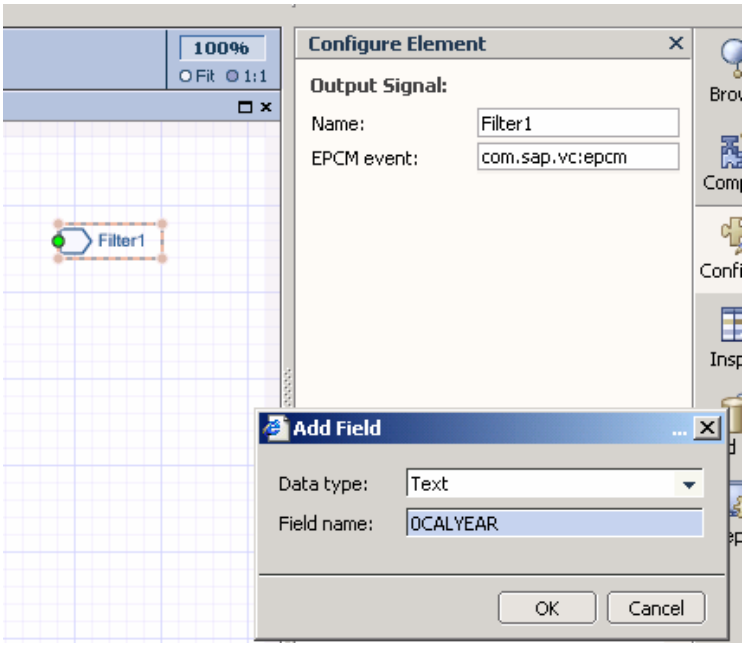
6. Add a Submit button to your form.



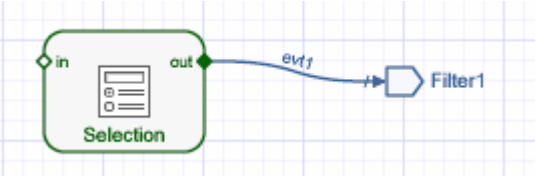
7. Add a Signal out item and make sure the name is “Filter1” and the namespace is specified to “com.sap.vc:epcm”. This is very important!!!



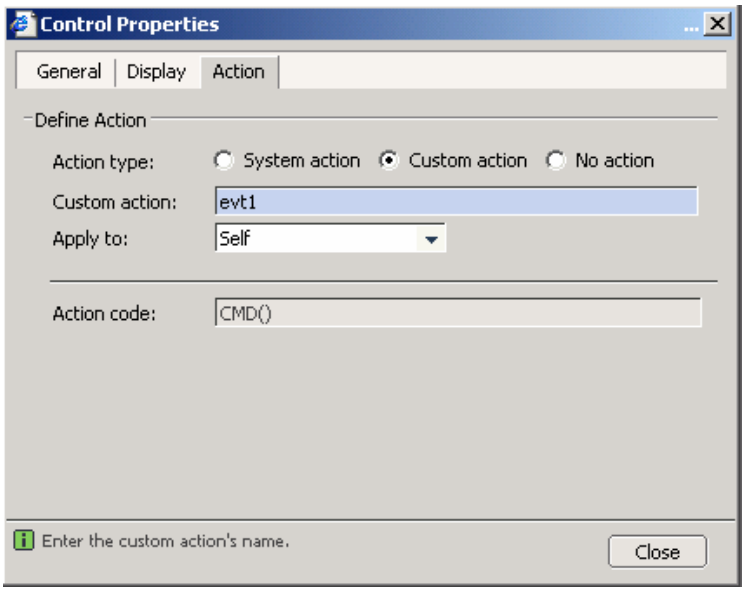
8. Add the field “0CALYEAR” to your signal out.



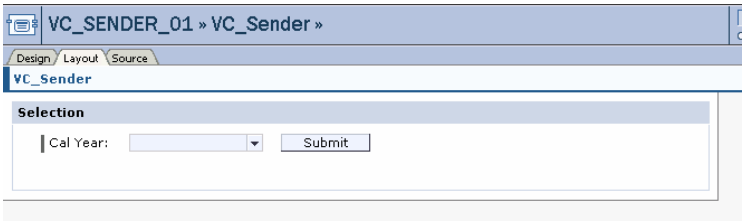
9. Connect your “Selection” form with the “Filter1” signal out …



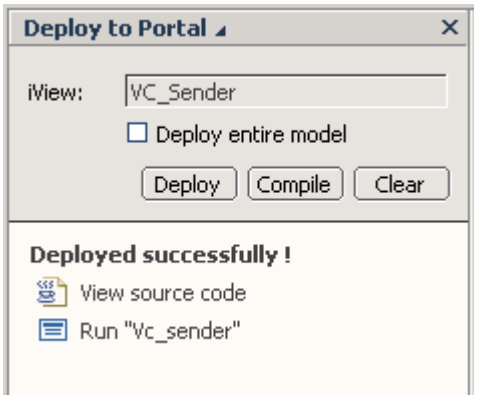
10. Update the properties of the submit button to call custom action “evt1”.



11. Adjust the layout of your VC Model.



12. Save and Deploy your model.



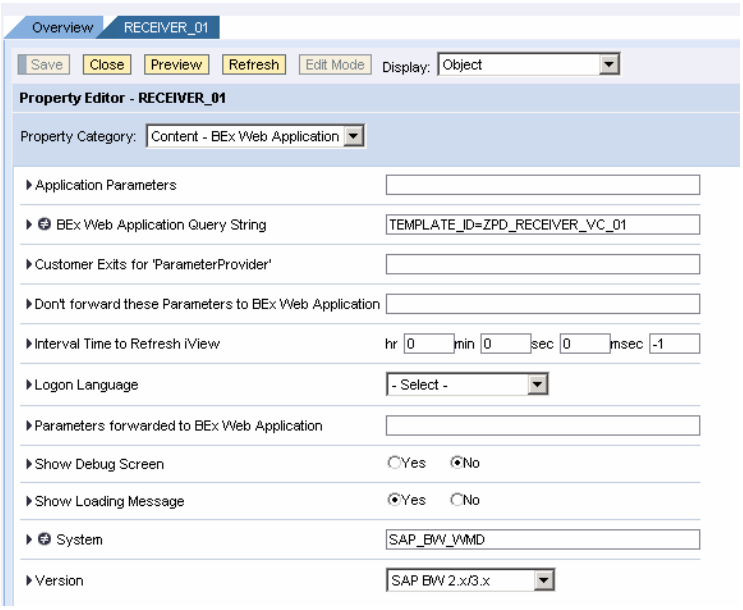
13. Launch the Web Application Designer (3.x) and build a web template that listens to this event and calls the web api based on this event.

    See the highlighted code to the right. This javascript function is subscribing to an event and using the Web API to filter to the value it receives from the event. The javascript function is converting the event specified in Filter1 to an xml document. We are reading the value in the xml document into the web api filter. See the Appendix for details on the format in which the event is received from Visual Composer Applications.

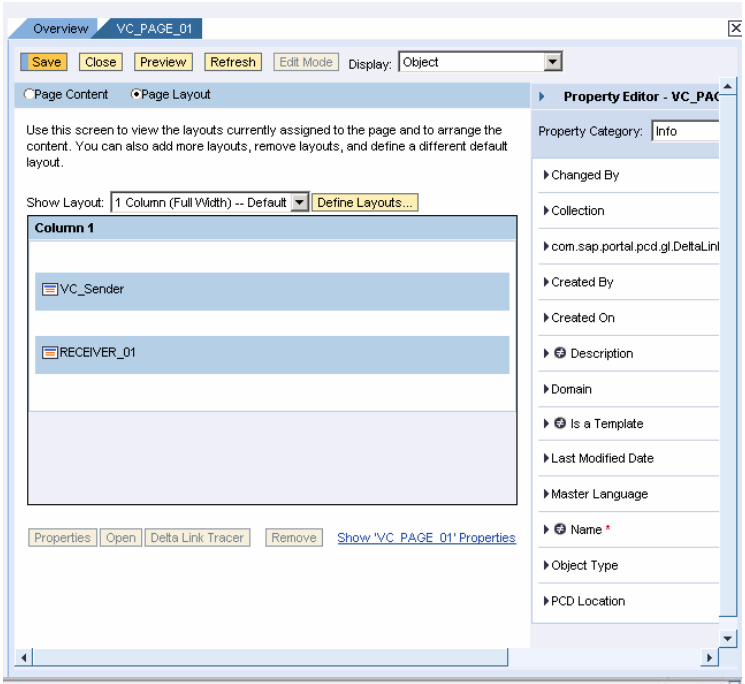
```
<!-- BW data source object tags -->
<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_DATA_PROVIDER"/>
      <param name="NAME" value="DATAPROVIDER_1"/>
      <param name="QUERY" value="PM_DX_C01_1"/>
      <param name="INFOCUBE" value="0D_DX_C01"/>
      DATA_PROVIDER:          DATAPROVIDER_1
</object>

<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_PROPERTIES"/>
      <param name="TEMPLATE_ID"
value="ZPD_RECEIVER_VC_01"/>
      TEMPLATE PROPERTIES
</object>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing
Control">
<TITLE>BW Web Application</TITLE>
    <link href="/sap/bw/Mime/BEx/StyleSheets/BWReports.css"
type="text/css" rel="stylesheet"/>
<script language=javascript>
 var xmldoc;

function load(xmlFile)
{
  // code for IE
  if (window.ActiveXObject)
  {
    xmldoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
    xmldoc.loadXML(xmlFile);
  }
  else
  {
    alert('Your browser cannot handle this script');
  }
}

    EPCMPROXY.subscribeEvent( "urn:com.sap.vc:epcm",  "Filter1",
window, "myreceiveEvent");
    function myreceiveEvent( eventObj ) {
            values = eventObj.dataObject;
            unesc_val = unescape(values);
            load(unesc_val);
            values =
xmldoc.getElementsByTagName("Row").item(0).getAttribute("F0CALYEA
R");
            var url = location.protocol
+"//"+location.host+SAP_BW_URL_Get()
                url = url +
"&data_provider=*&multi=X&FILTER_IOBJNM=0CALYEAR&FILTER_VA
LUE=" + values +"&filter_collapse=";
                SAPBWOpenURL(url);
              }
            </script>

</HEAD>
<BODY>
<P><object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="GET_ITEM"/>
      <param name="NAME" value="TABLE_1"/>
      <param name="ITEM_CLASS"
```

```
value="CL_RSR_WWW_ITEM_GRID"/>
      <param name="DATA_PROVIDER" value="DATAPROVIDER_1"/>
      ITEM:          TABLE_1
</object></P>
</BODY>

</HTML>
```

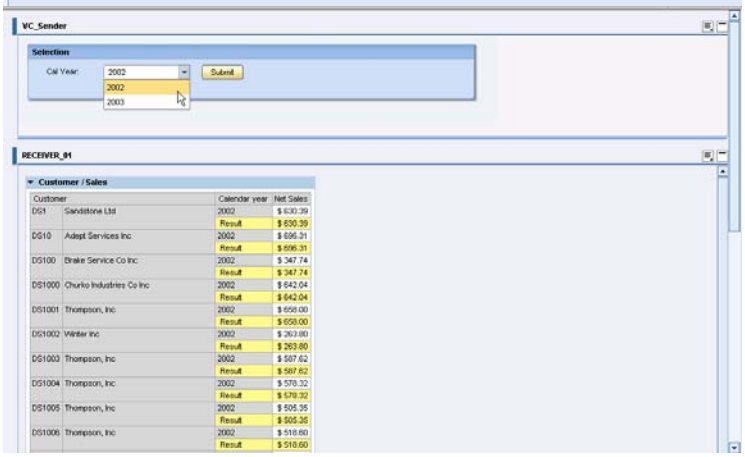
14. Build an iView for your BI Web Application.



15. Build a page on the Portal using delta link iViews. Ensure that the Visual Composer iView and your Web Application Designer iView are in this page. Save the page.



16. Now your parameters are being passed from VC to your web application!!! Keep in mind that this example is just passing one parameter. You can pass multiple values across from Visual Composer to your Web Application on a single signal out. This is demonstrated later in this paper.



4.2 Sending Events from Web Application Designer to Visual Composer

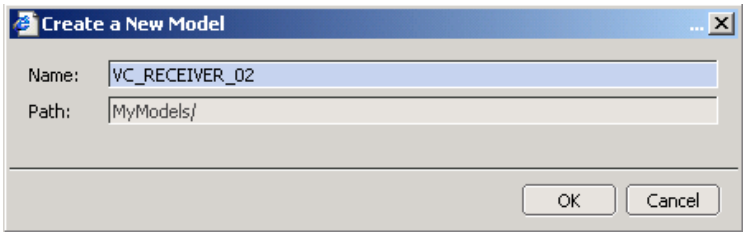
1. Launch the Web Application Designer (3.x) and build a web template that sends an event to Visual Composer.

   NOTE: This event is named "Filter1" and needs to be named the same event within Visual Composer. Also, make sure the field specified in extvalue to the right is the field within Visual Composer's Signal In …

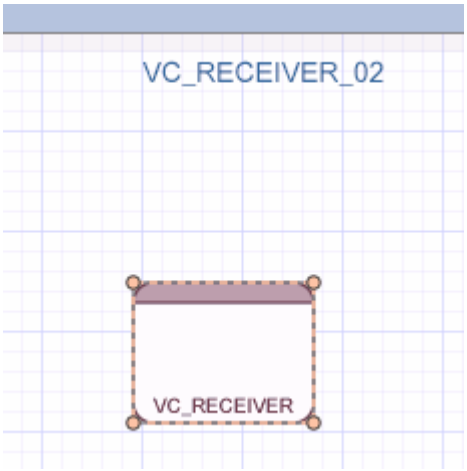
   See the highlighted code to the right. This javascript function is sending an event and the format is specified in the format needed for Visual Composer Applications. This is being set by the dropdown list for 0CALYEAR.

```
<!-- BW data source object tags -->
<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_DATA_PROVIDER"/>
      <param name="NAME" value="DATAPROVIDER_1"/>
      <param name="QUERY" value="PM_DX_C01_1"/>
      <param name="INFOCUBE" value="0D_DX_C01"/>
      DATA_PROVIDER:          DATAPROVIDER_1
</object>

<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_PROPERTIES"/>
      <param name="TEMPLATE_ID"
value="ZPD_SENDER_WAD_01"/>
      TEMPLATE PROPERTIES
</object>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing
Control">
<TITLE>BW Web Application</TITLE>
    <link href="/sap/bw/Mime/BEx/StyleSheets/BWReports.css"
type="text/css" rel="stylesheet"/>
<script language=javascript>
<!--
        function raiseEvents(value) {
           extvalue = "<Params version='2' ><Row F0CALYEAR='" +
value + "' /></Params>";
           EPCMPROXY.raiseEvent( "urn:com.sap.vc:epcm", "Filter1",
extvalue, null );
    }
-->
</script></HEAD>
<BODY>
<P>
<TABLE cellSpacing=1 cellPadding=1 width="75%" border=0>

 <TR>
   <TD vAlign=top>
<form name="myform">
<select name="myselect" size="1"
onchange="raiseEvents(document.myform.myselect.options[document.m
yform.myselect.selectedIndex].value);">
<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="GET_ITEM"/>
      <param name="NAME" value="DROPDOWNBOX_1"/>
      <param name="ITEM_CLASS"
value="CL_RSR_WWW_ITEM_FILTER_DDOWN"/>
      <param name="DATA_PROVIDER" value="DATAPROVIDER_1"/>
      <param name="GENERATE_CAPTION" value=""/>
      <param name="IOBJNM" value="0CALYEAR"/>
      <param name="ONLY_VALUES" value="X"/>
      ITEM:          DROPDOWNBOX_1
</object>
</select>
</form>
</TD></TR>
 <TR>
   <TD vAlign=top></TD></TR></TABLE></P>
</BODY>
</HTML>
```

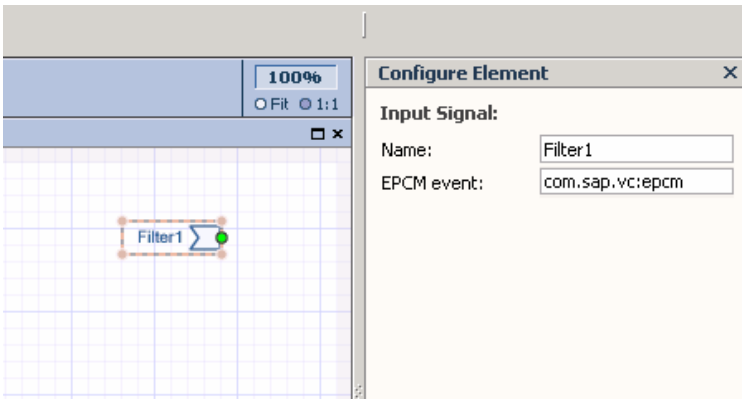
2. Create a new model within VC.



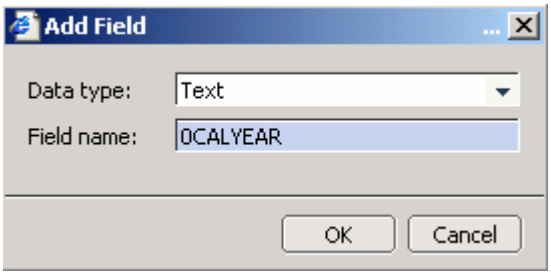
3. Drag and drop an iView within your model and name iView "VC_RECEIVER".



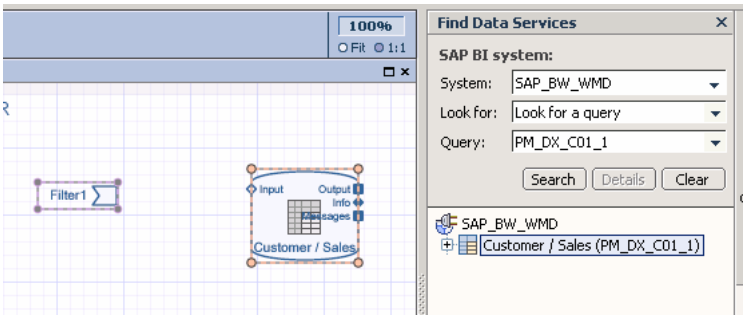
4. Add a signal in called “Filter1” on namepace “com.sap.vc:epcm”



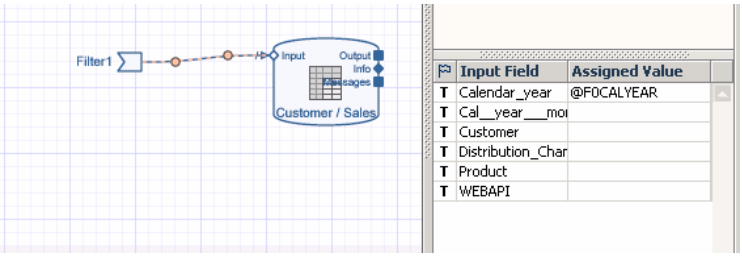
5. Add a field called “0CALYEAR” to your signal in.



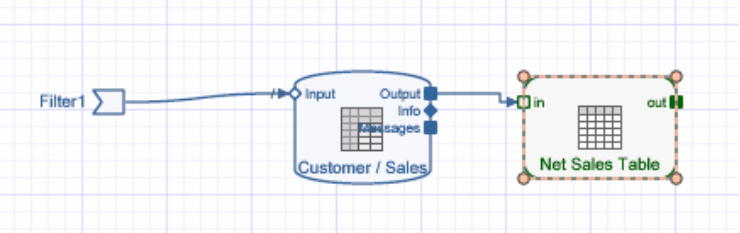
6. Add a data service that you want to pass this filter to …



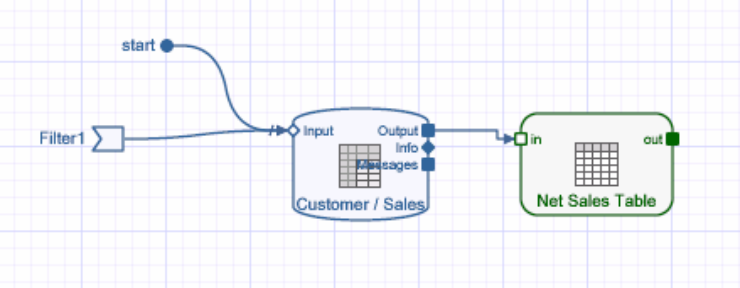
7. Connect the input signal to your data service and specify the mapping for the field you added.



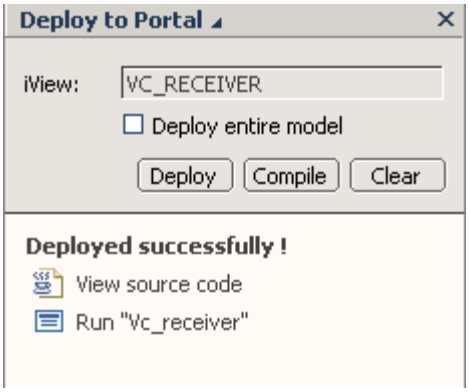
8. Add a table to display the results of the data service and specify your fields within the table.



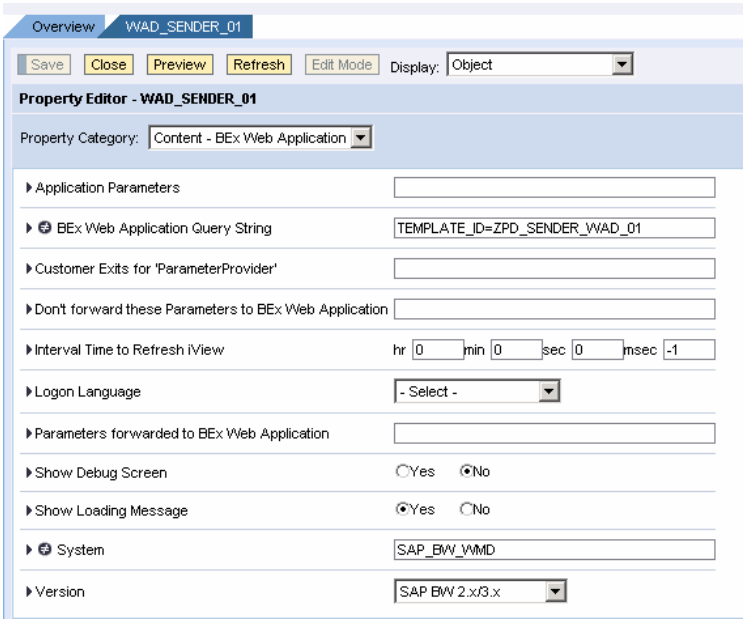
9. Add a start point to your model.



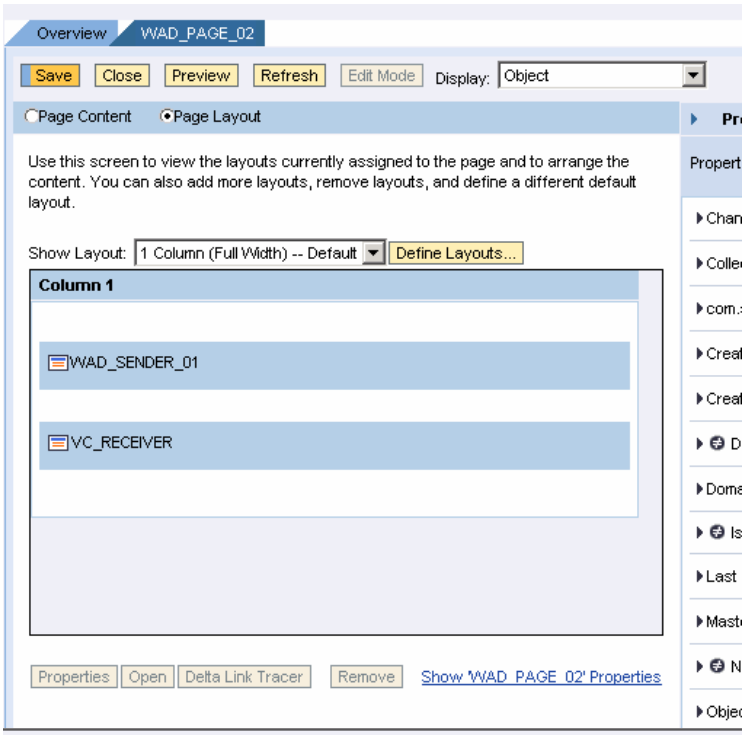
10. Save and deploy your model.



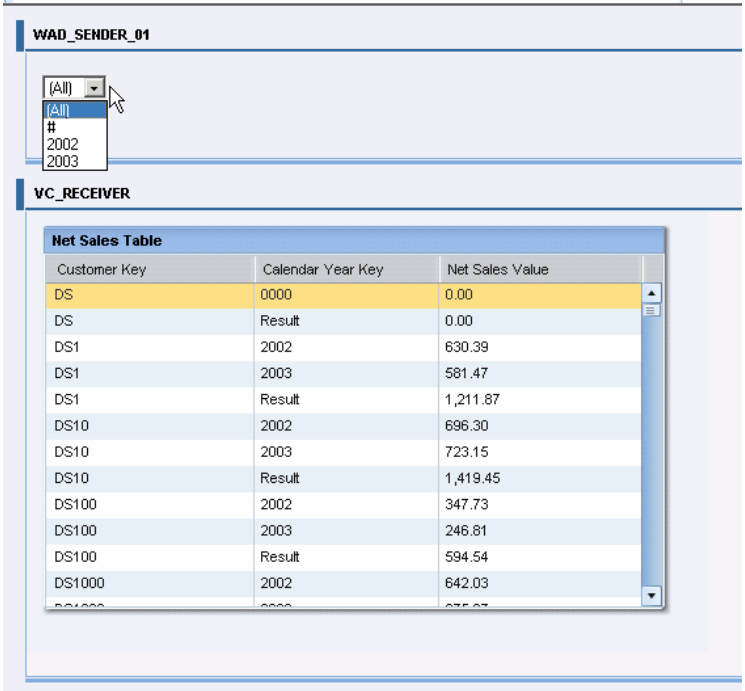
11. Build an iView for your BI Web Application.



12. Build a page on the Portal using delta link iViews. Ensure that the Visual Composer iView and your Web Application Designer iView are in this page. Save the page.



13. Now your parameters are being passed from Web Application Designer to your Visual Composer application!!! Keep in mind that this example is just passing one parameter. You can pass multiple values across from Web Application Designer to Visual Composer on a single signal out.

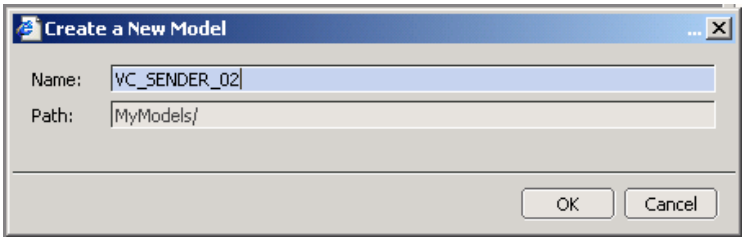


4.3 Dynamic Events from Visual Composer to Web Application Designer

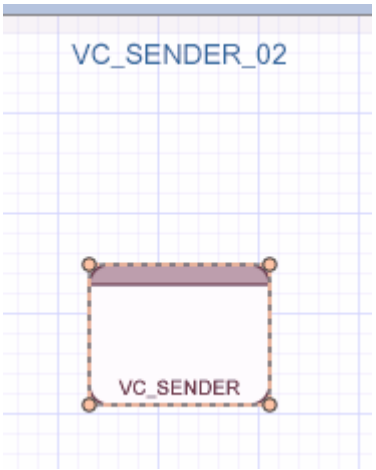
This solution using javascript within the Web Application Designer to read all filter values passed from Visual Composer and dynamically builds the web api string for filtering. This requires your Visual Composer to name your objects on your signal out with the technical name of the

infoobjects you want to filter on. The Web Application Designer will read this metadata to generate the web api for filtering.

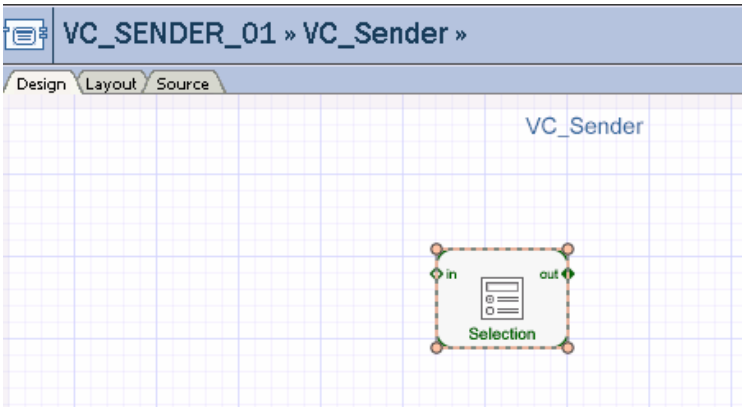
1. Create a New Model within VC.



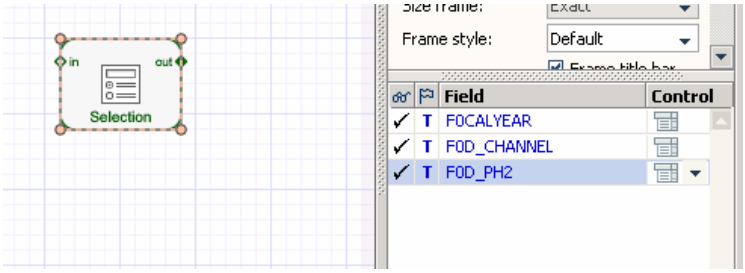
2. Drag and drop an iView within your model and name iView "VC_SENDER".



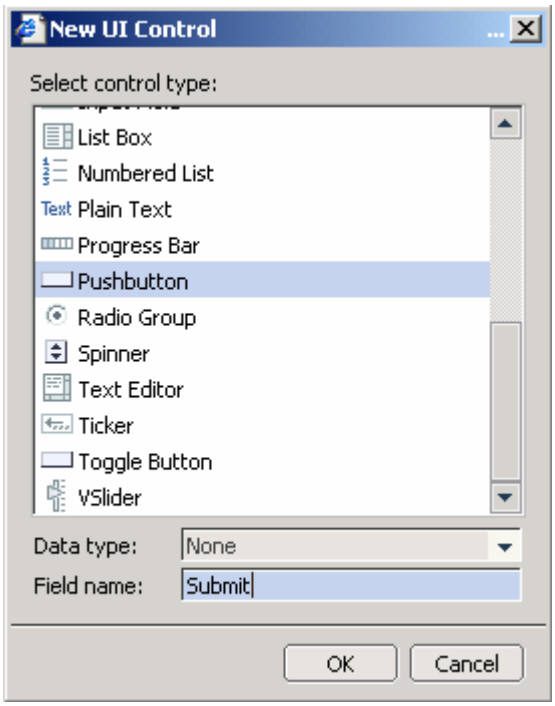
3. Drag and drop a form within your model and name this form "Selection".



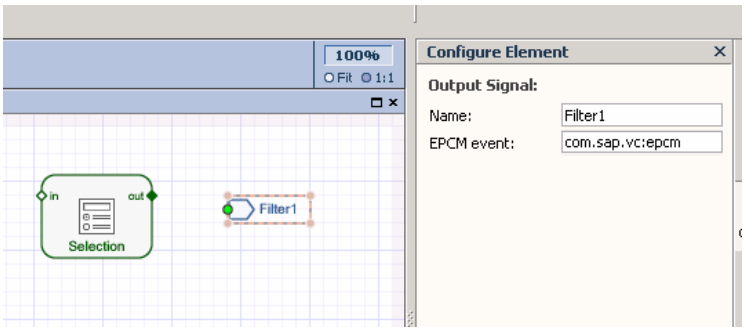
4. Add 3 fields that are defined off of dropdown lists based off either static or dynamic entry lists (could be based off characteristic).



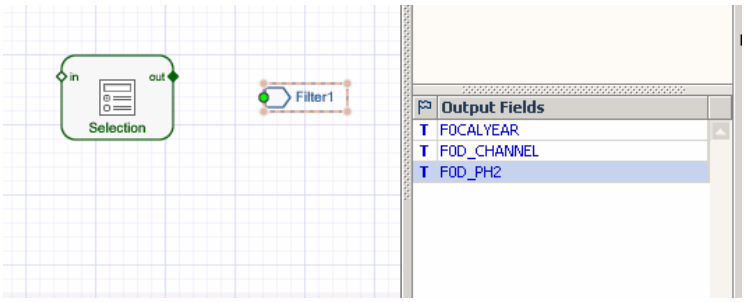
5. Add a Submit button to your form.



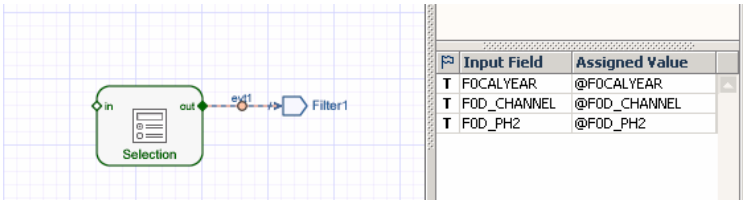
6. Add a Signal out item and make sure the name is “Filter1” and the namespace is specified to “com.sap.vc:epcm”. This is very important!!!



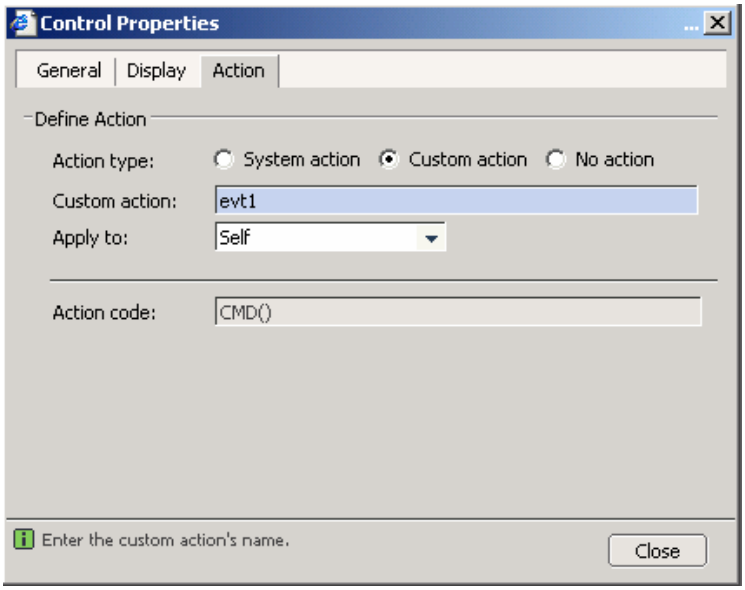
7. Add your 3 fields to the output signal. Make sure these fields are named after the technical name of an infoobject. Also, keep in mind that objects that start with zero automatically have the character “F” added to the front.



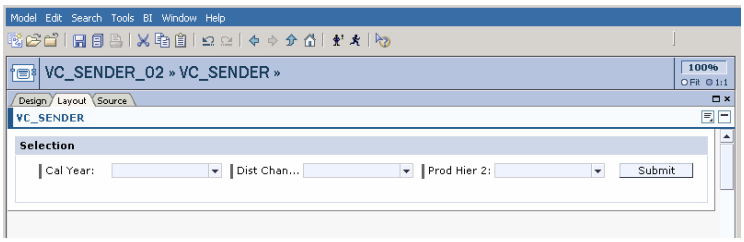
8. Connect your “Selection” form with the “Filter1” signal out … Ensure that the mapping is done.



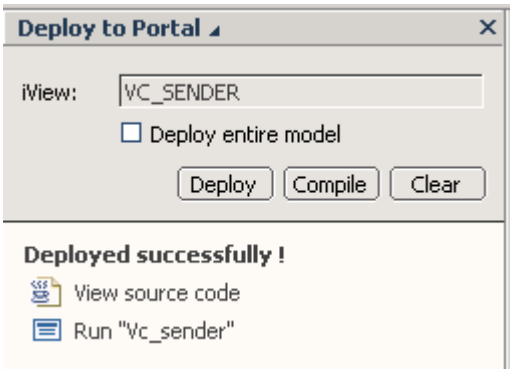
9. Update the properties of the submit button to call custom action “evt1”.



10. Adjust the layout of your VC Model.



11. Save and Deploy your model.



12. Launch the Web Application Designer (3.x) and build a web template that listens to this event and calls the web api based on this event.

    NOTE: This code doesn't deal with removing of filters or choosing the "all" value in a dropdown. This will need to be added... This will be based upon what your value for "All" is defined as in your entry list within Visual Composer.

    See the highlighted code to the right. This javascript function is subscribing to an event and using the Web API to

```
<!-- BW data source object tags -->
<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_DATA_PROVIDER"/>
      <param name="NAME" value="DATAPROVIDER_1"/>
      <param name="QUERY" value="PM_DX_C01_1"/>
      <param name="INFOCUBE" value="0D_DX_C01"/>
      DATA_PROVIDER:          DATAPROVIDER_1
</object>

<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_PROPERTIES"/>
      <param name="TEMPLATE_ID"
value="ZPD_RECEIVER_VC_02"/>
      TEMPLATE PROPERTIES
</object>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing
```

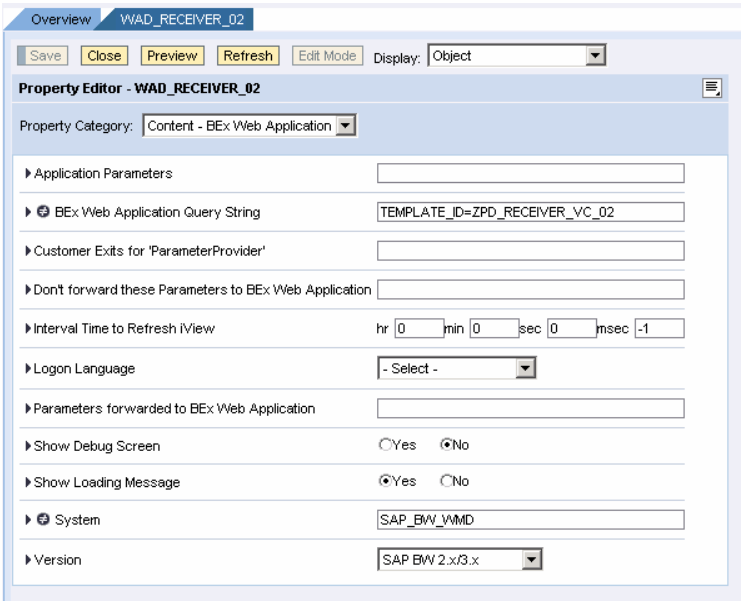
filter to the value it receives from the event. The javascript function is receiving the value specified in Filter1 through an xml document. See the Appendix for details on the format in which the event is received from Visual Composer Applications.

Keep in mind that this requires that Visual Composer names the fields based on infoobject technical names in your output signal from Visual Composer.

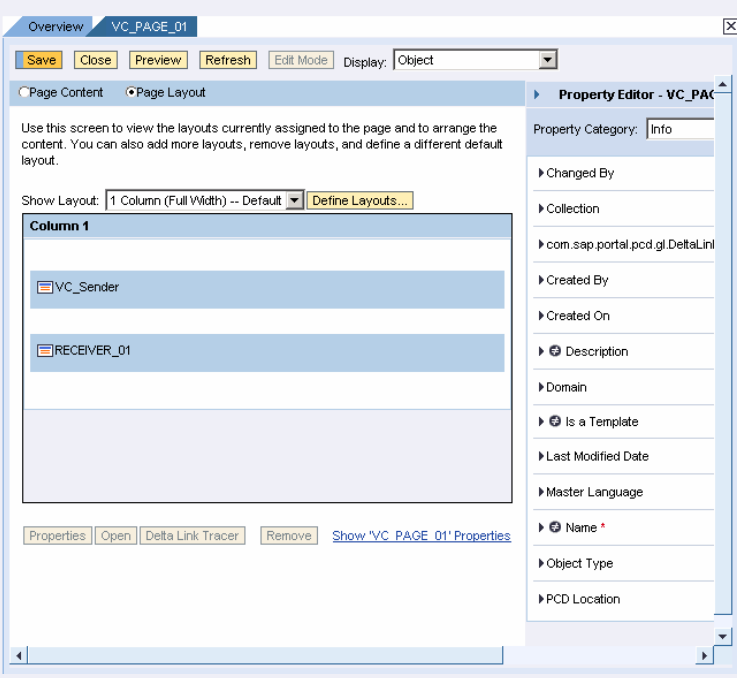
```
Control">
<TITLE>BW Web Application</TITLE>
    <link href="/sap/bw/Mime/BEx/StyleSheets/BWReports.css"
type="text/css" rel="stylesheet"/>
<script language=javascript>
var xmldoc;

function load(xmlFile)
{
  // code for IE
  if (window.ActiveXObject)
  {
    xmldoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
    xmldoc.loadXML(xmlFile);
  }
  else
  {
    alert('Your browser cannot handle this script');
  }
}
EPCMPROXY.subscribeEvent( "urn:com.sap.vc:epcm",  "Filter1",
window, "myreceiveEvent");

function myreceiveEvent( eventObj )
{
  values = eventObj.dataObject;
  unesc_val = unescape(values);
  load(unesc_val);
  len_val =
xmldoc.getElementsByTagName("Row").item(0).attributes.length;
  var valueArray = new Array();
  valueArray =
xmldoc.getElementsByTagName("Row").item(0).attributes;
  var append_url = "";
  var j = 1;
  for(i = 0; i < len_val; i++)
  {
    tempArr1 = valueArray[i].xml.split('=');
    IOBJ = tempArr1[0];
    VAL =
xmldoc.getElementsByTagName("Row").item(0).getAttribute(IOBJ);

    if (IOBJ.charAt(0) == 'F')
    {
      IOBJ = IOBJ.substring(1, IOBJ.length);
    }
    if (VAL != "")
    {
      append_url = append_url + "&FILTER_IOBJNM_" + j + "=" + IOBJ +
"&FILTER_VALUE_" + j + "=" + VAL;
      j++;
    }
  }
  var url = location.protocol +"//"+location.host+SAP_BW_URL_Get()
  url = url + "&data_provider=*&multi=X" + append_url
+"&filter_collapse=";
  SAPBWOpenURL(url);
}
</script>

</HEAD>
<BODY>
<P><object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="GET_ITEM"/>
      <param name="NAME" value="TABLE_1"/>
      <param name="ITEM_CLASS"
```

```
value="CL_RSR_WWW_ITEM_GRID"/>
      <param name="DATA_PROVIDER" value="DATAPROVIDER_1"/>
      ITEM:          TABLE_1
</object></P>
</BODY>
</HTML>
```

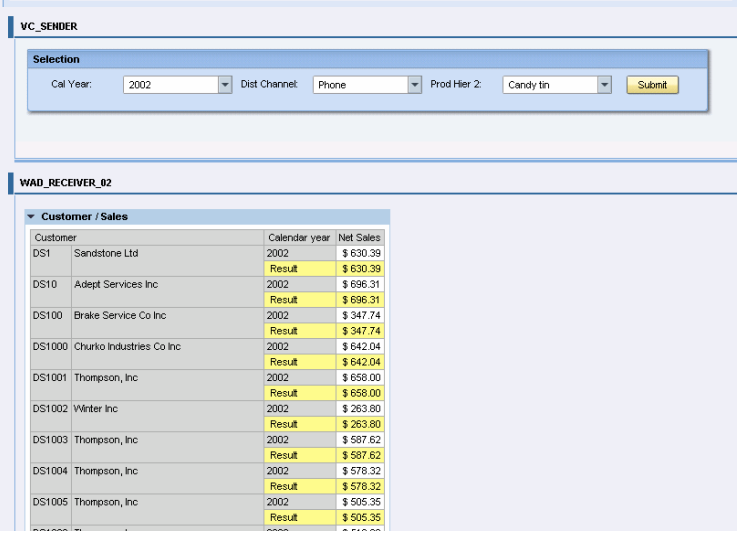
13. Build an iView for your BI Web Application.



14. Build a page on the Portal using delta link iViews. Ensure that the Visual Composer iView and your Web Application Designer iView are in this page. Save the page.



15. Now your parameters are being passed from VC to your web application!!! This example is passing multiple values across from Visual Composer to your Web Application on a single signal out.



4.4 Modularize Web Application Development with eventing between multiple web templates

1. Launch the Web Application Designer (3.x) and build a web template that sends events.

   See the highlighted code to the right. This javascript function is raising an event and passing a value from the dropdown for 0CALYEAR to this event. The javascript function is parsing out the exact values from the event specified in Filter1.

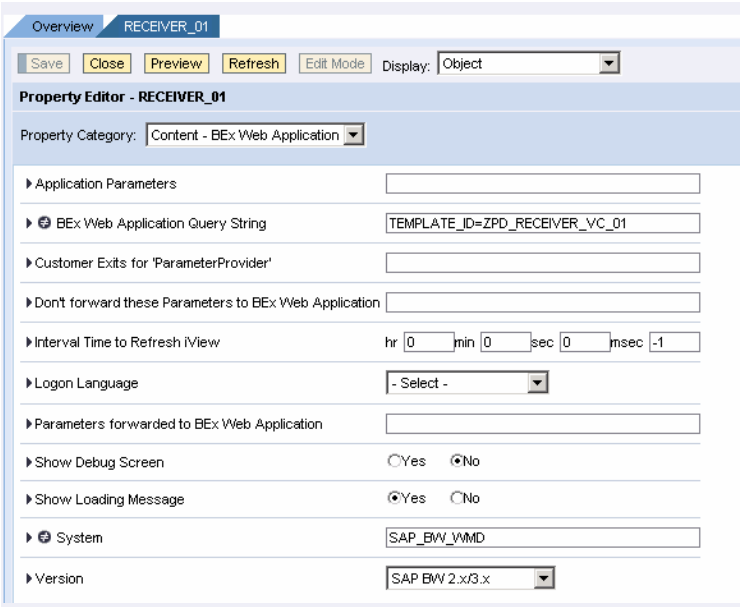
```
<!-- BW data source object tags -->
<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_DATA_PROVIDER"/>
      <param name="NAME" value="DATAPROVIDER_1"/>
      <param name="QUERY" value="PM_DX_C01_1"/>
      <param name="INFOCUBE" value="0D_DX_C01"/>
      DATA_PROVIDER:          DATAPROVIDER_1
</object>

<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_PROPERTIES"/>
      <param name="TEMPLATE_ID"
value="ZPD_SENDER_WAD_01"/>
      TEMPLATE PROPERTIES
</object>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing
Control">
<TITLE>BW Web Application</TITLE>
    <link href="/sap/bw/Mime/BEx/StyleSheets/BWReports.css"
type="text/css" rel="stylesheet"/>
<script language=javascript>
         function raiseEvents(value) {
           EPCMPROXY.raiseEvent( "urn:com.sap.vc:epcm", "Filter1",
value, null );
    }
</script>
</HEAD>
<BODY>
<P>
<TABLE cellSpacing=1 cellPadding=1 width="75%" border=0>

 <TR>
  <TD vAlign=top>
<form name="myform">
<select name="myselect" size="1"
onchange="raiseEvents(document.myform.myselect.options[document.m
yform.myselect.selectedIndex].value);">
<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="GET_ITEM"/>
      <param name="NAME" value="DROPDOWNBOX_1"/>
      <param name="ITEM_CLASS"
value="CL_RSR_WWW_ITEM_FILTER_DDOWN"/>
      <param name="DATA_PROVIDER" value="DATAPROVIDER_1"/>
      <param name="GENERATE_CAPTION" value=""/>
      <param name="IOBJNM" value="0CALYEAR"/>
      <param name="ONLY_VALUES" value="X"/>
      ITEM:          DROPDOWNBOX_1
</object>
</select>
</form>
</TD></TR>
 <TR>
  <TD vAlign=top></TD></TR></TABLE></P>
</BODY>
</HTML>
```

2. Build a web template within the web application designer that listens to events broadcast by the template built in step 1.

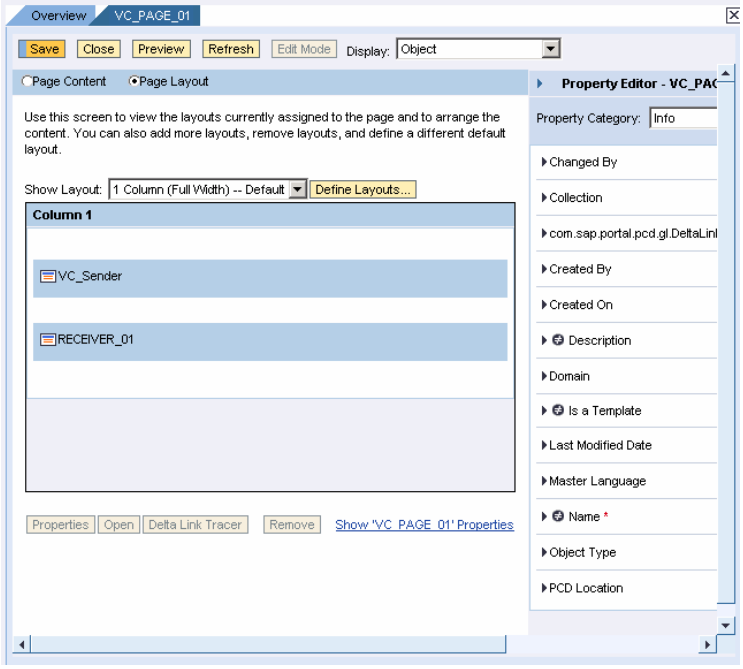
   See the highlighted code to the right. This javascript function is subscribing to an event and using the Web API to filter to the value it receives from the event. Since the format being passed in on the event is the exact value, we can pass this value directly into the Web API command.

```
<!-- BW data source object tags -->
<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_DATA_PROVIDER"/>
      <param name="NAME" value="DATAPROVIDER_1"/>
      <param name="QUERY" value="PM_DX_C01_1"/>
      <param name="INFOCUBE" value="0D_DX_C01"/>
      DATA_PROVIDER:          DATAPROVIDER_1
</object>

<object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="SET_PROPERTIES"/>
      <param name="TEMPLATE_ID"
value="ZPD_RECEIVER_WAD_01"/>
      TEMPLATE PROPERTIES
</object>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing
Control">
<TITLE>BW Web Application</TITLE>
    <link href="/sap/bw/Mime/BEx/StyleSheets/BWReports.css"
type="text/css" rel="stylesheet"/>
      <script language=javascript>
    EPCMPROXY.subscribeEvent( "urn:com.sap.vc:epcm",  "Filter1",
window, "myreceiveEvent");
    function myreceiveEvent( eventObj ) {
            values = eventObj.dataObject;
            var url = location.protocol
+"//"+location.host+SAP_BW_URL_Get()
                url = url +
"&data_provider=*&multi=X&FILTER_IOBJNM=0CALYEAR&FILTER_VA
LUE=" +values +"&filter_collapse=";
                 SAPBWOpenURL(url);
                }
             </script>
</HEAD>
<BODY>
<P><object>
      <param name="OWNER" value="SAP_BW"/>
      <param name="CMD" value="GET_ITEM"/>
      <param name="NAME" value="TABLE_1"/>
      <param name="ITEM_CLASS"
value="CL_RSR_WWW_ITEM_GRID"/>
      <param name="DATA_PROVIDER" value="DATAPROVIDER_1"/>
      ITEM:          TABLE_1
</object></P>
</BODY>
</HTML>
```

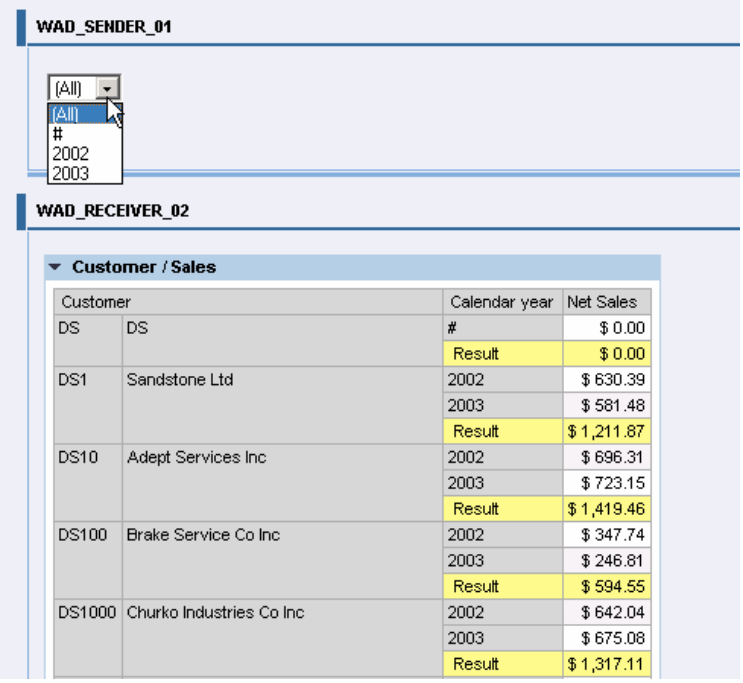
3. Build iViews for each of your BI Web Applications.



4. Build a page on the Portal using delta link iViews. Ensure that both Web Application Designer iViews are in this page. Save the page.



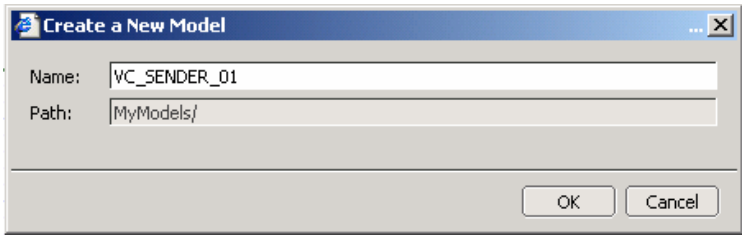
5. Now your parameters are being passed from your BI web applications!!! This will help for building web applications with very modular pieces. For example, this top iView can set the Fiscal Year Filter for all iViews in your web application. If you navigate to new reports, they will also be updated with this filter and this filter will not be reset!!!



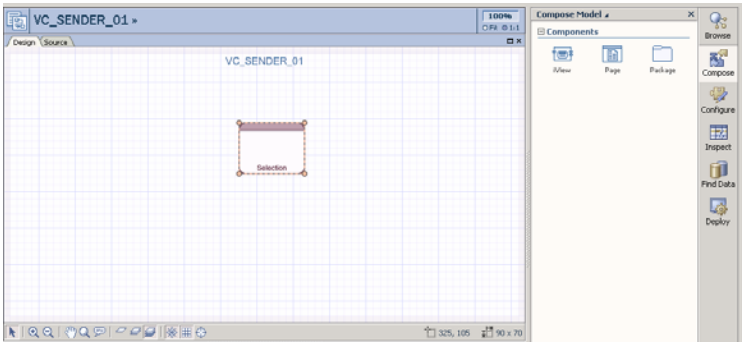
Customer		Calendar year	Net Sales
DS	DS	#	$ 0.00
		Result	$ 0.00
DS1	Sandstone Ltd	2002	$ 630.39
		2003	$ 581.48
		Result	$ 1,211.87
DS10	Adept Services Inc	2002	$ 696.31
		2003	$ 723.15
		Result	$ 1,419.46
DS100	Brake Service Co Inc	2002	$ 347.74
		2003	$ 246.81
		Result	$ 594.55
DS1000	Churko Industries Co Inc	2002	$ 642.04
		2003	$ 675.08
		Result	$ 1,317.11

4.5 Modularize Visual Composer Development with eventing between multiple Visual Composer applications!!!

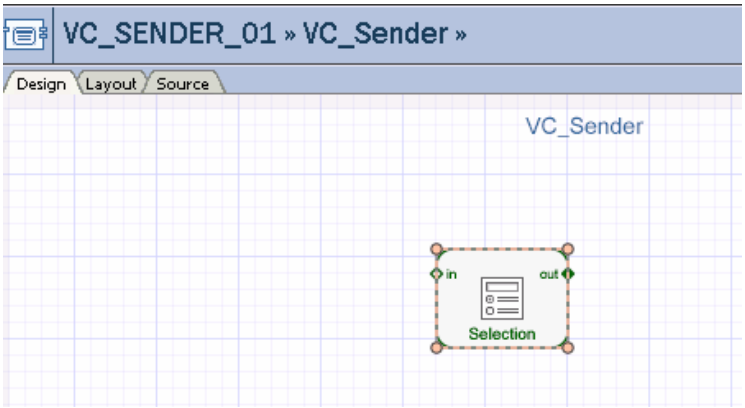
1. Create a New Model within VC.



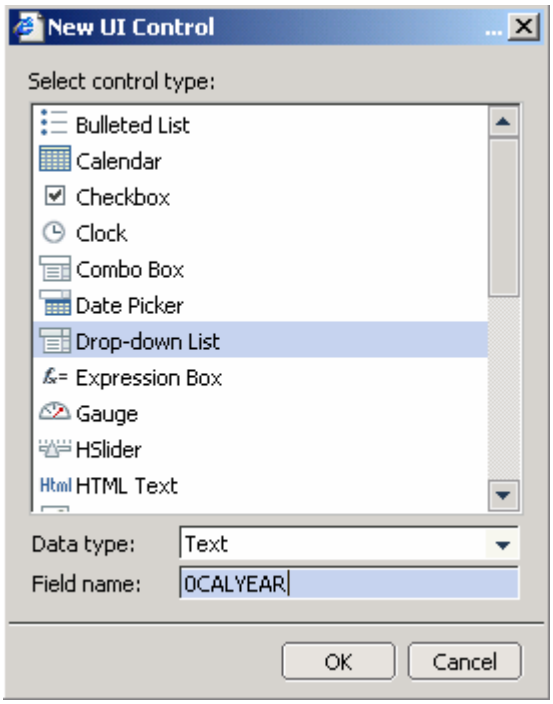
2. Drag and drop an iView within your model and name iView “VC_Sender”.



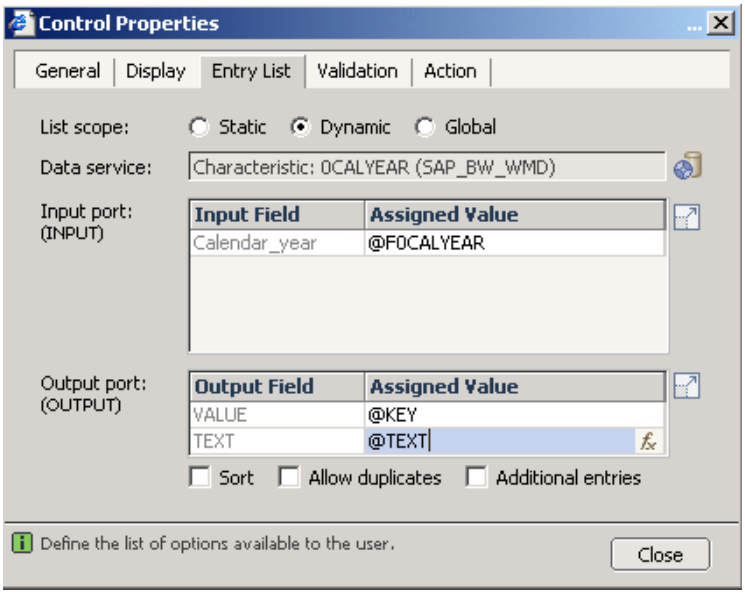
3. Drag and drop a form within your model and name this form “Selection”.



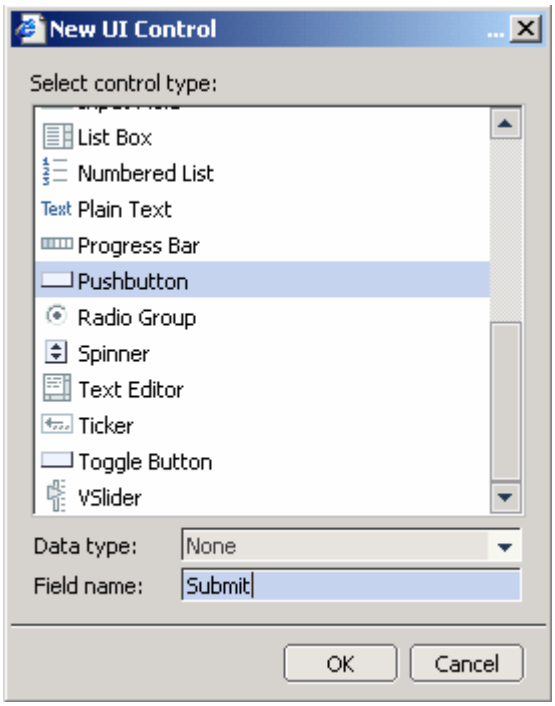
4. Add a dropdown list within your form and name this field “0CALYEAR”. Keep in mind that fields cannot start with “0” within VC and the field will automatically be renamed to “F0CALYEAR”.



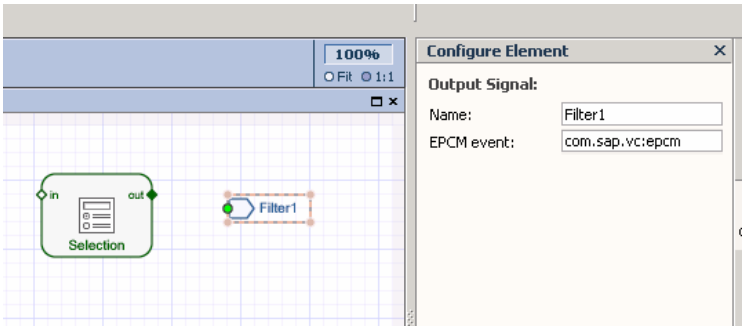
5. Specify either a static or dynamic dropdown list in the properties of this object.



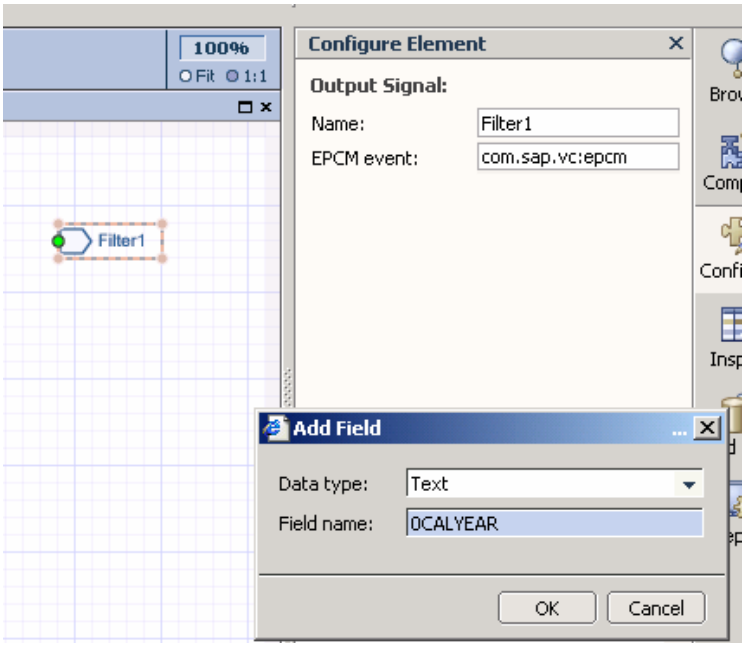
6. Add a Submit button to your form.



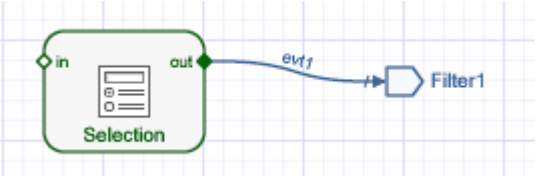
7. Add a Signal out item and make sure the name is “Filter1” and the namespace is specified to “com.sap.vc:epcm”. This is very important!!!



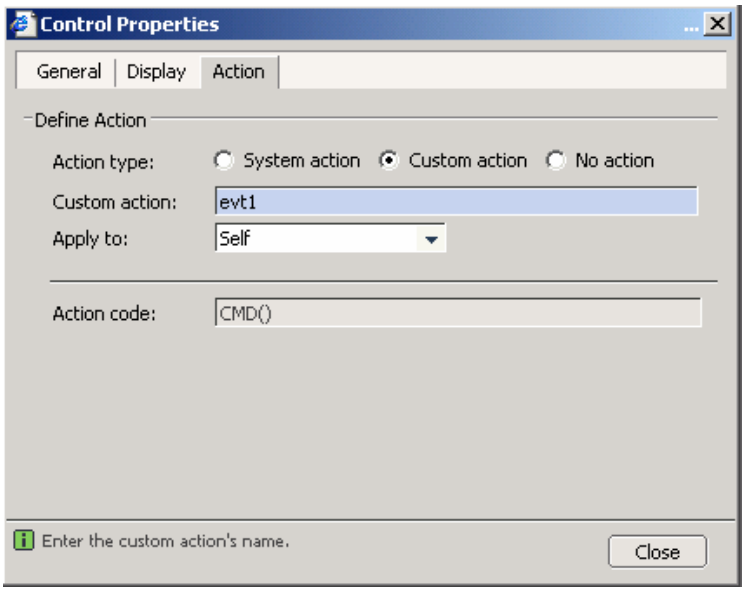
8. Add the field “0CALYEAR” to your signal out.



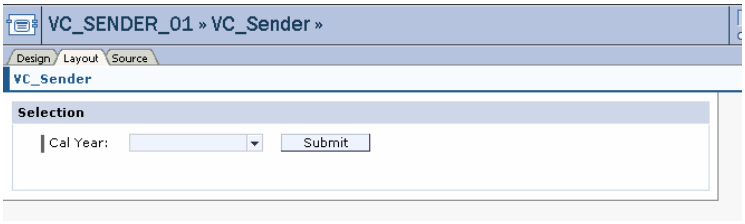
9. Connect your “Selection” form with the “Filter1” signal out …



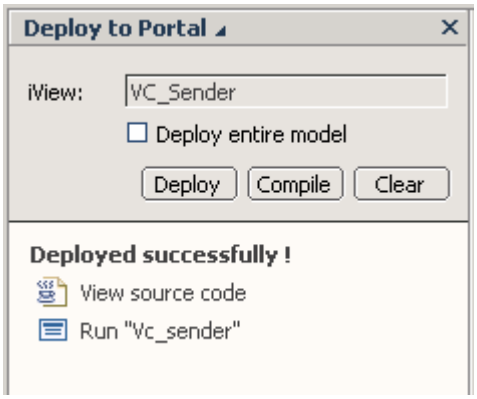
10. Update the properties of the submit button to call custom action “evt1”.



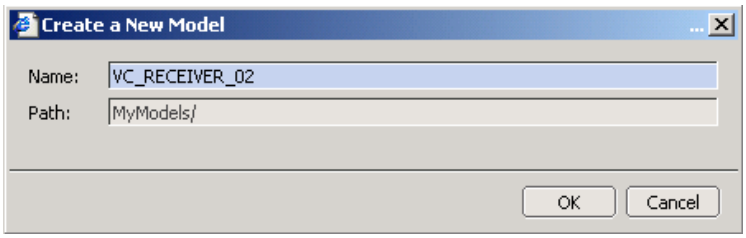
11. Adjust the layout of your VC Model.



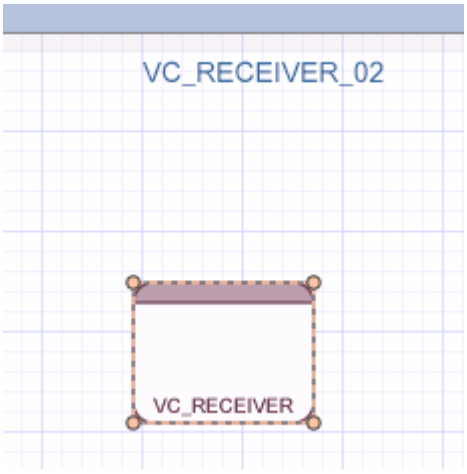
12. Save and Deploy your model.



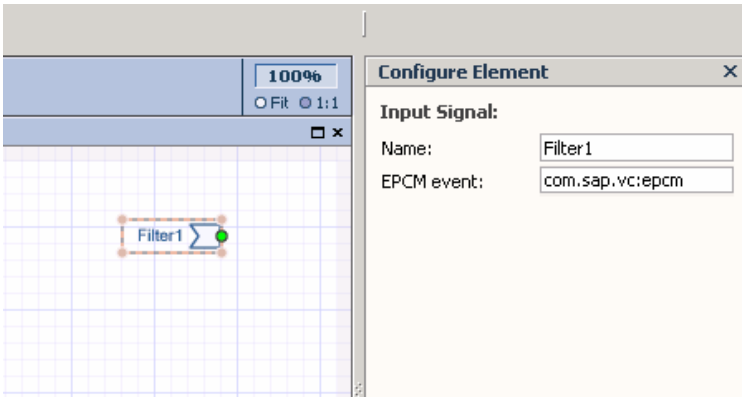
13. Create a new model within VC.



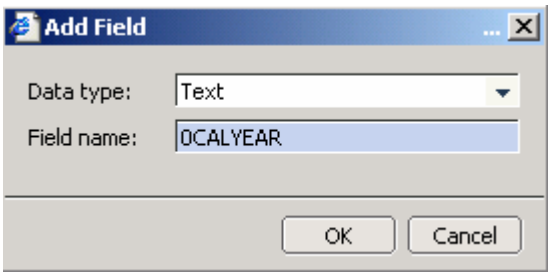
14. Drag and drop an iView within your model and name iView "VC_RECEIVER".



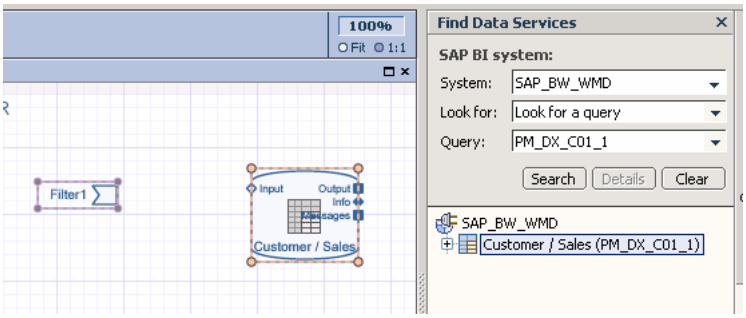
15. Add a signal in called "Filter1" on namepace "com.sap.vc:epcm"



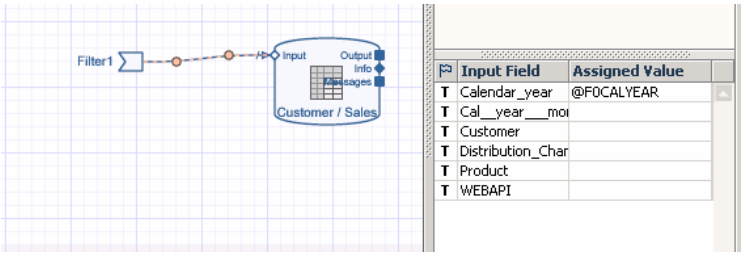
16. Add a field called “0CALYEAR” to your signal in.



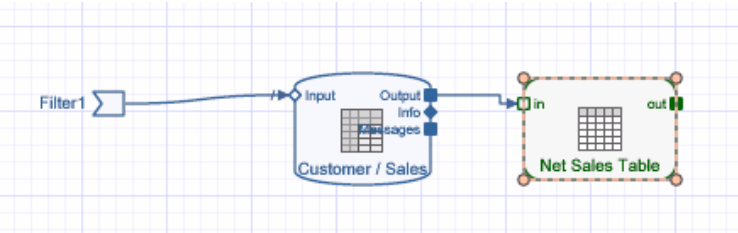
17. Add a data service that you want to pass this filter to …



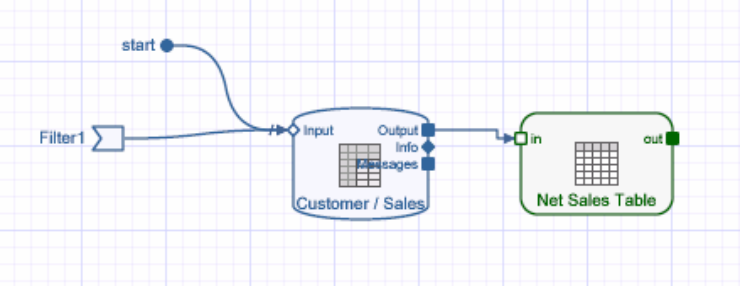
18. Connect the input signal to your data service and specify the mapping for the field you added.



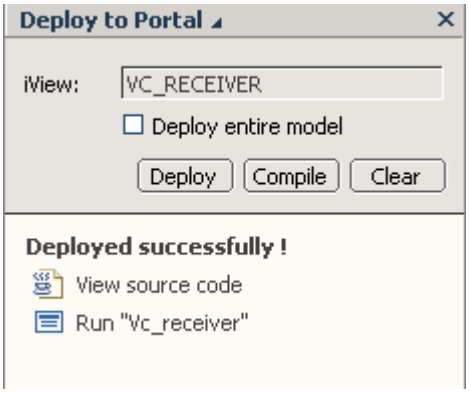
19. Add a table to display the results of the data service and specify your fields within the table.



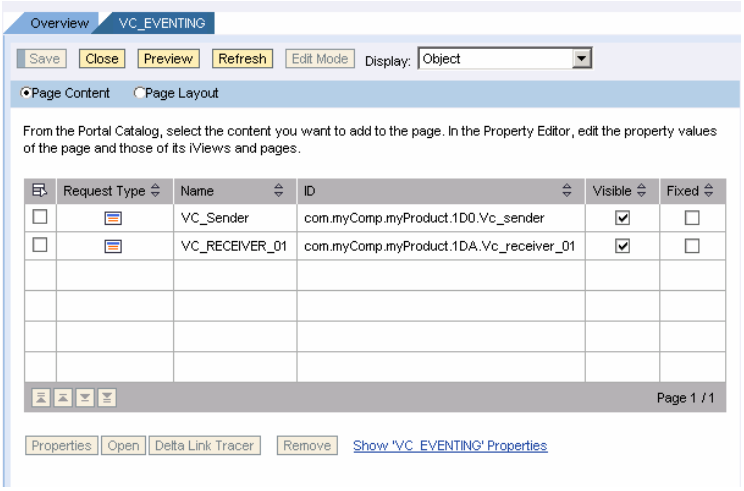
20. Add a start point to your model.



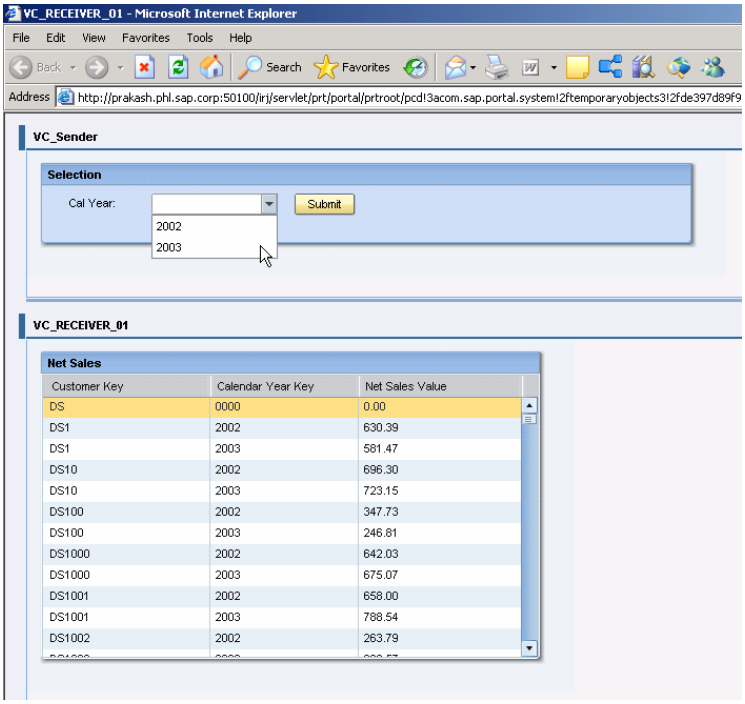
21. Save and deploy your model.



22. Build a portal page that has both of these iViews.



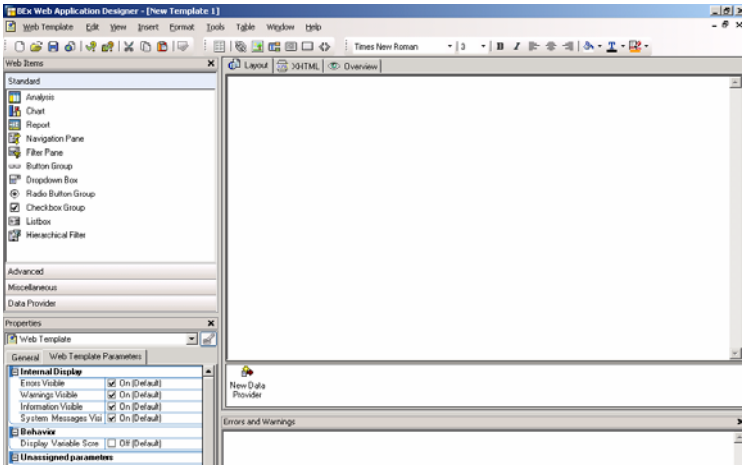
23. Run your Portal Page and see the modular components!!!



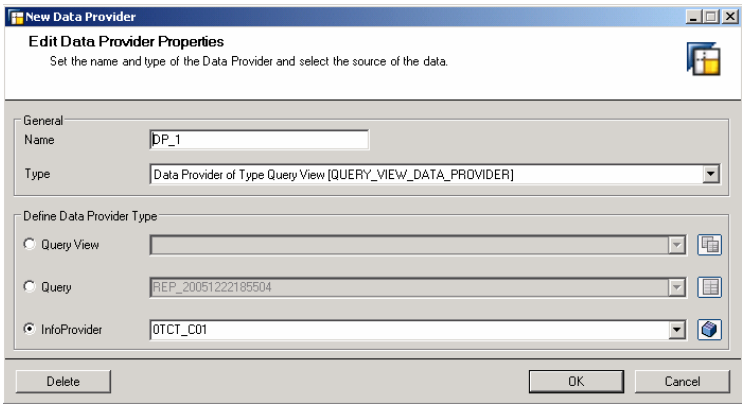
4.6 Working with Javascript within the SAP NetWeaver 2004s Web Application Designer

As of SAP NetWeaver 2004s SP7, you can use javascript within your web applications using the new web application designer. Therefore, web applications built with either the 3.x or 2004s web application designers can subscribe to events as demonstrated below.

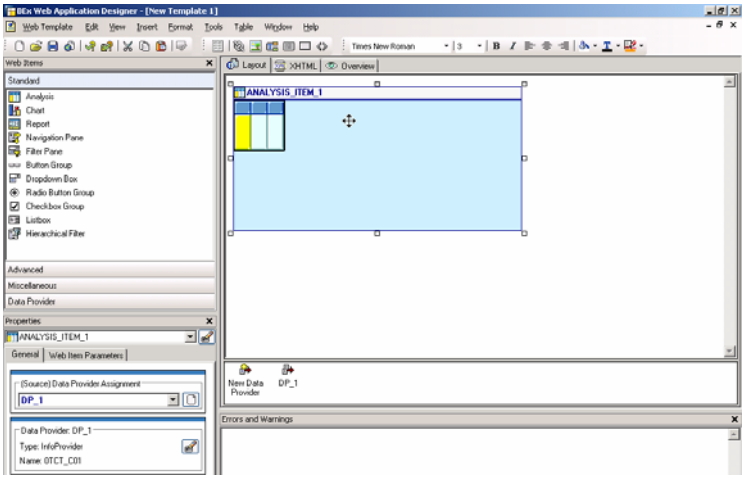
1. Launch the new web application designer.



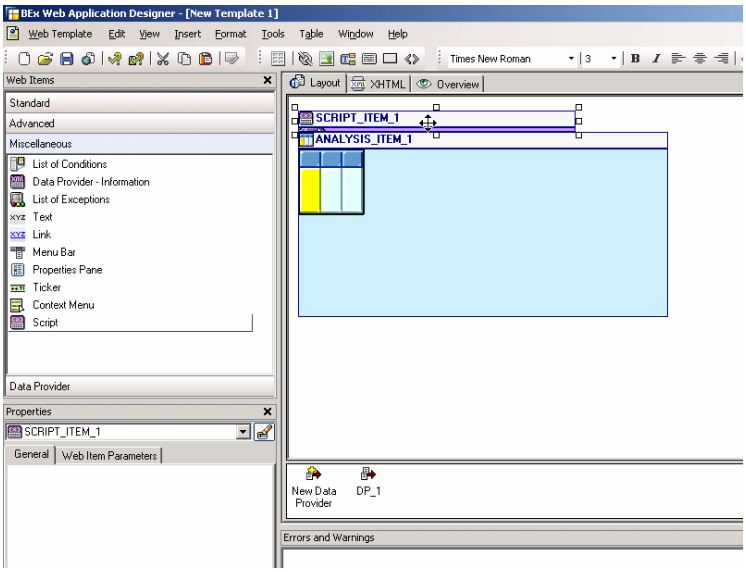
2. Create a new DataProvider.



3. Drag and drop the "Analysis Item" into your web application.

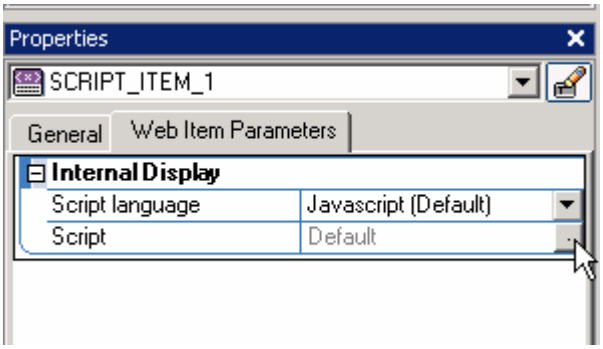


4. Drag and drop the “Script” item into your web template.

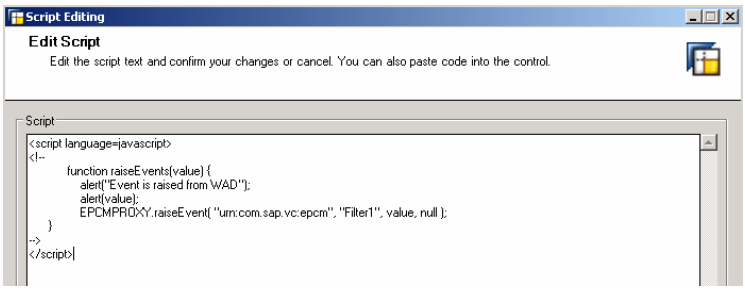


5. Within the web item properties, choose the button to enter your javascript code.

   NOTE: You cannot add javascript directly to the XHTML in your web application. You must use this web item!



6. Enter your code to subscribe or broadcast events to integration with Visual Composer or other web applications.

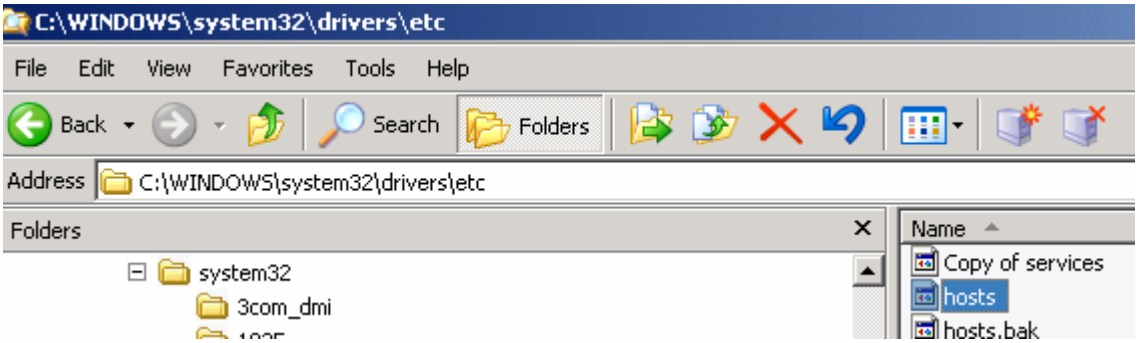


5 Common Issues and Resolutions

5.1 Domain Check

ISSUE: Javascript error: EPCMPROXY undefined.

RESOLUTION: Ensure that your Portal and BI system are within the same domain. This is not just the top level domain, but includes the fully qualified domain name. Eventing isn't supported cross domain … If need be, you can update your hosts file at



to specify that these systems are in the same domain. For example:

CASE 1: Eventing Doesn't work:

BI System: cdphl819.phl.sap.corp
Portal: http://CHIN00419027A.chi.sap.corp:50100/irj/portal

CASE 2: Eventing Does work:

BI System: cdphl819.phl.sap.corp
Portal: http://CHIN00419027A.phl.sap.corp:50100/irj/portal

Since I was setup in CASE 1, I made an entry into my hosts file such as this:

127.0.0.1 CHIN00419027A.phl.sap.corp localhost

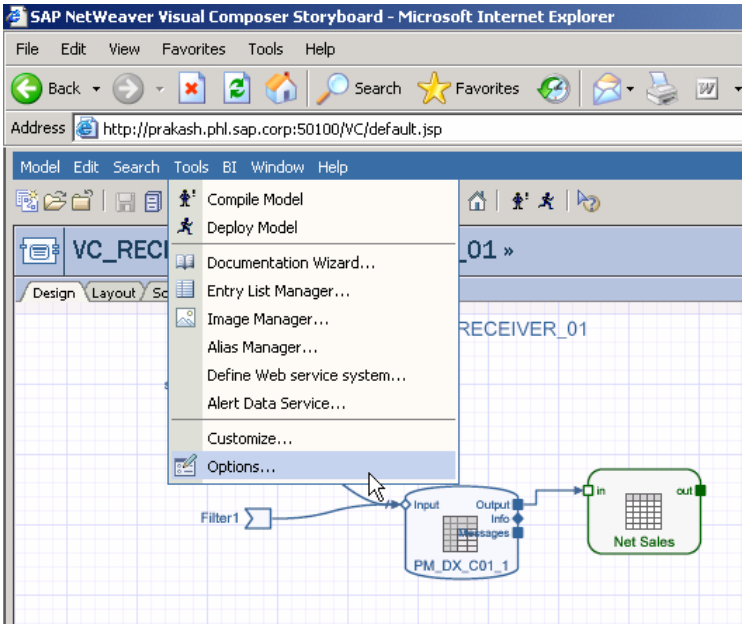
Because this is client side eventing, this hosts file entry would need to be specified on the CLIENT side. If it's on the server you would need to register a fully qualified domain name in the same domain for both systems and have users access these url's from the same domain.

6 Troubleshooting

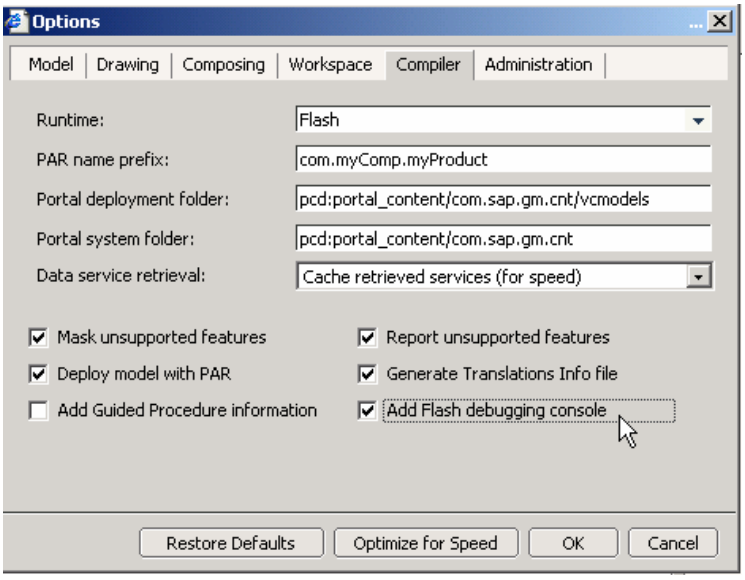
6.1 Flash Debugger

As of SAP NetWeaver 2004s SP7, Visual Composer allows you to use the flash debugger. This will expose all actions and events within an individual Visual Composer application.

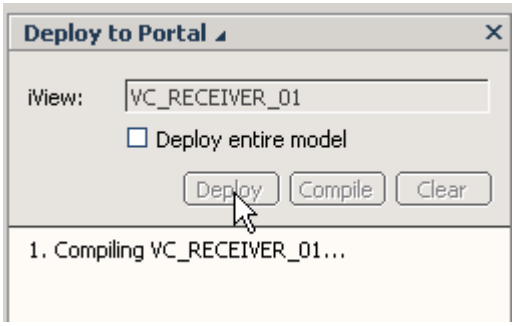
1. Within Visual Composer, go to Tools -> Options.



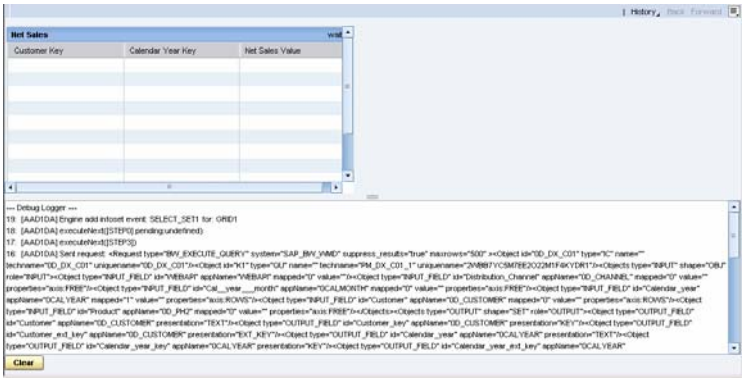
2. Go to the Compiler tab. Choose the "Add Flash debugging console" option and choose OK.



3. Deploy your Visual Composer application.



4. Run your VC Application. You will now see the Flash Debug Logger which will show you all events taking place at runtime!!!



6.2 Alerts within Javascript

Because we are working with multiple applications, as we broadcast or send out events, to ensure this occurs properly, you can use alerts within javascript

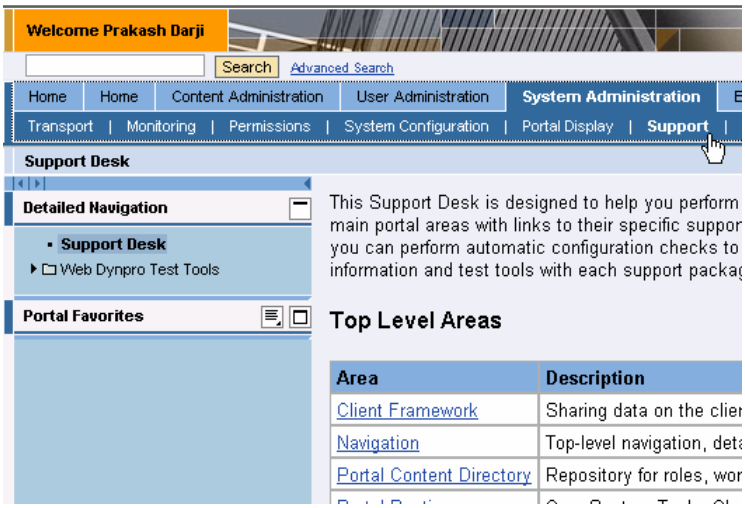
1. Within your javascript functions, alert out values to trace your values you are raising or subscribing within events.

```
<script language=javascript>
<!--
        function raiseEvents(value) {
           alert("Event is raised from WAD");
           alert("Sending out Filter1 value = " + value);
           EPCMPROXY.raiseEvent( "urn:com.sap.vc:epcm", "Filter1",
value, null );
     }
-->
</script>
```

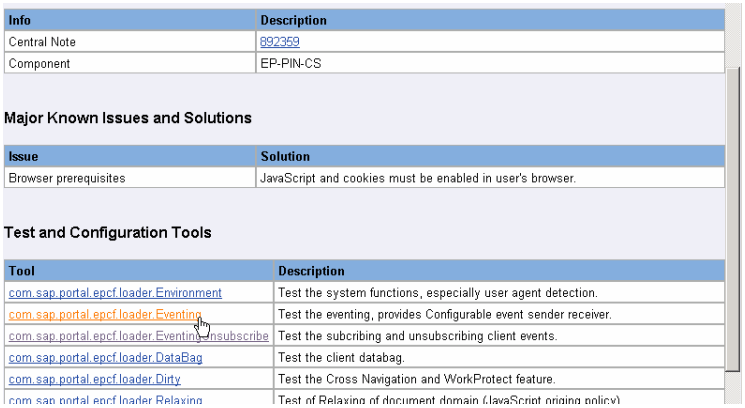
6.3 Portal Eventing Support Framework

The Enterprise Portal provides a framework to register events for troubleshooting issues with eventing between iViews.

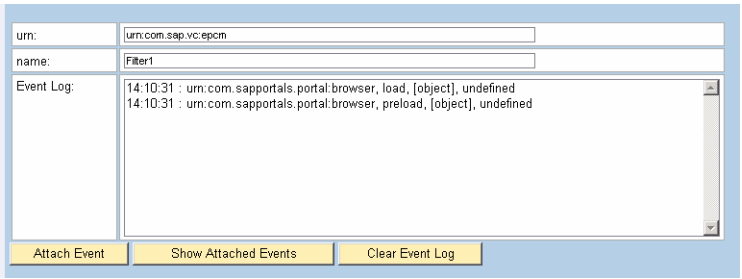
1. Login to the Enterprise Portal and navigate to "System Administration" -> "Support" -> "Client Framework".



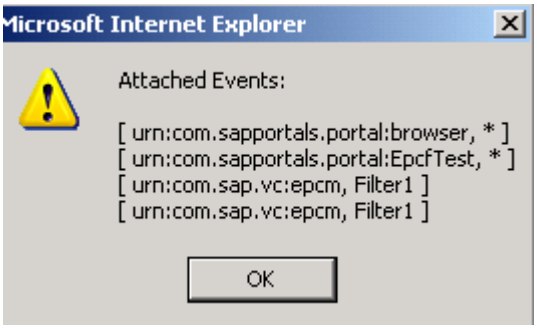
2. Choose Eventing to register your events.



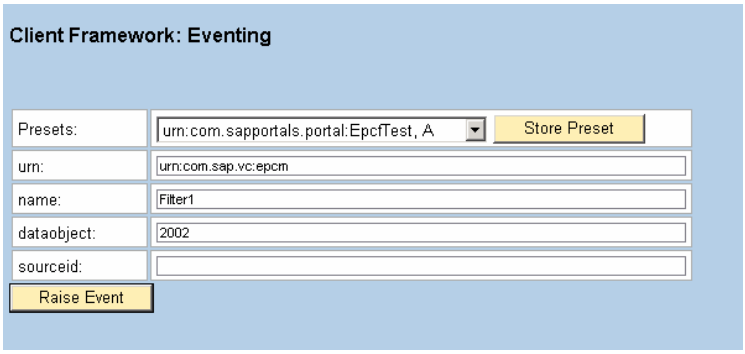
3. Enter your namespace and event and attach event.



4. Click “Show Attached Events” and ensure that your event is registered.



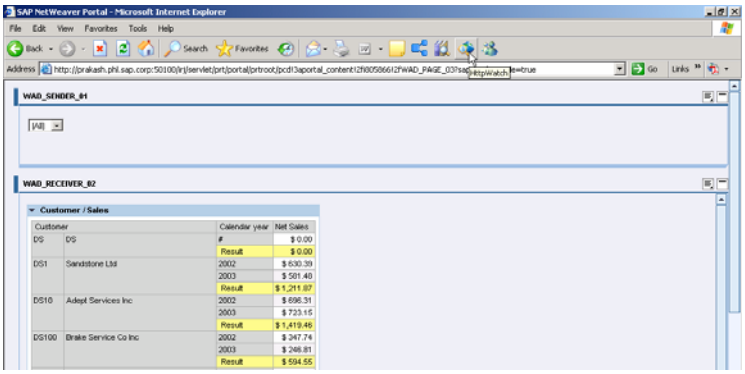
5. Enter your namespace, and event and raise the event and ensure it is registered.



6.4 HttpWatch

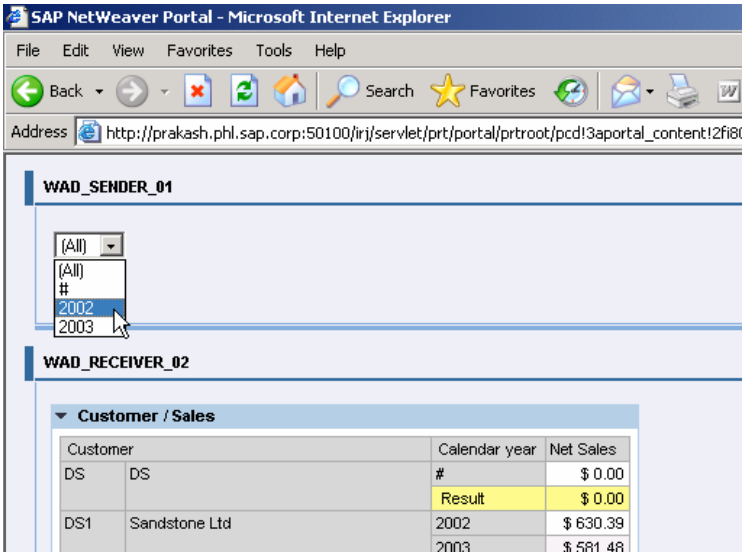
HttpWatch allows you to track html calls. This requires that this software is installed. Also, keep in mind that this doesn't offer you internal commands to flash and will help with eventing between portal iViews, but not off much assistance within the flash runtime. The Flash Debugger should be used for the flash runtime. HttpWatch will help immensely for web application designer dashboards.

1. Run your web application and turn on the HTTPWATCH with the icon on your toolbar.

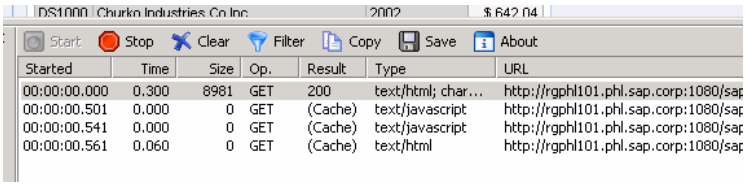


2. Choose Start to log http calls.

3. Trigger an event that will update other iViews.



4. All URL calls are logged and you can use this to ensure that filter values are passed properly …



7 Appendix

Visual Composer will send out events in ascii code format. For example,

1 field in signal out from VC:
Field 1: F0FISCPER
Field Value 1: 2002001

ASCII Code format:

```
%3CParams%20version%3D%222%22%20%3E%3CRow%20F0FISCPER%3D%222002001%22%20/%3E%
3C/Params%3E
```


HTML Code format:
```
< Params version = "2" > <Row F0FISCPER="2002001" /></Params>
```

3 fields from signal out within VC:
Field 1: F0FISCPER
Field Value 1: 2002001
Field 2: F0CALYEAR
Field Value 2: 2002
Field 3: F0COMP_CODE
Field Value 3: 1000

ASCII Code format:

```
%3CParams%20version%3D%222%22%20%3E%3CRow%20F0FISCPER%3D%222002%22%20F0CALYEA
R%3D%222002%22%20F0COMP_CODE%3D%221000%22%20/%3E%3C/Params%3E
```


HTML Code format:
```
< Params version = "2" > <Row F0FISCPER="2002001" F0CALYEAR="2002"
F0COMP_CODE="1000" /></Params>
```

Note that fields within Visual Composer cannot start with zero. Therefore, if you name a field starting with "0", an "F" is automatically put in front of this field. It is recommended to name fields with the technical name of infoobjects on the output signal. This way, you can build dynamic api commands within the web application designer.

Also, note that I make use of xml documents and the javascript DOM to handle reading these xml documents.

Also, make sure you implement most of your javascript as includes within your web application and store these includes centrally on the Mime repository. This will allow you to be flexible and scale changes without having to update every web template manually (for example, adding another infoobject to filter on can be updated in every web application in mass) …

For a tutorial on XML DOM objects, see http://www.w3schools.com/dom/default.asp

www.sap.com/netweaver