



**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**

© Copyright 2005 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C<sup>®</sup>, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data

contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

SAP NetWeaver "How-to" Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.

# **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

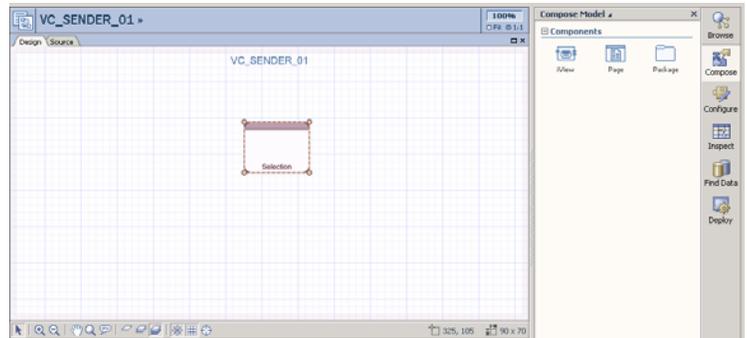
#### 4.1 Sending Events from Visual Composer to Web Application Designer

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.

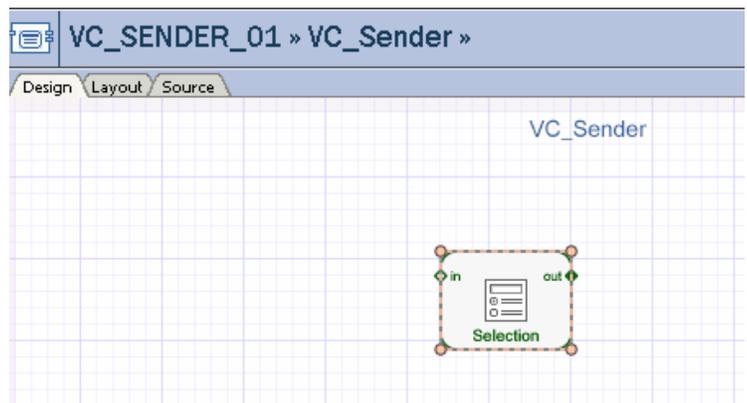
1. Create a New Model within VC.



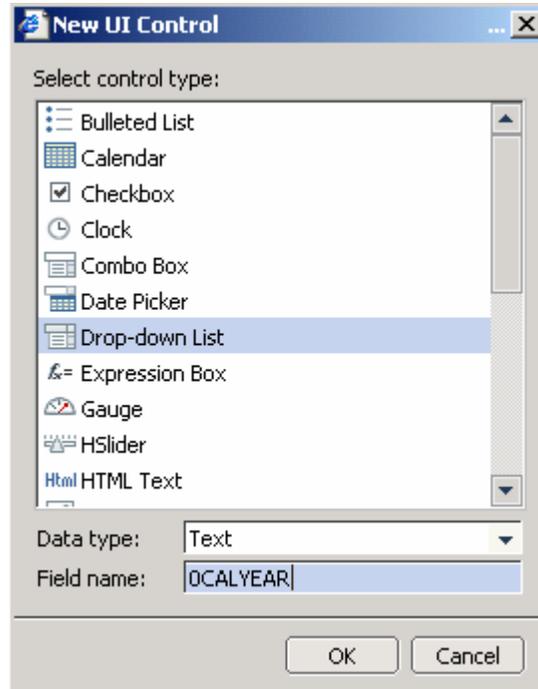
2. Drag and drop an iView within your model and name iView "VC\_Sender".



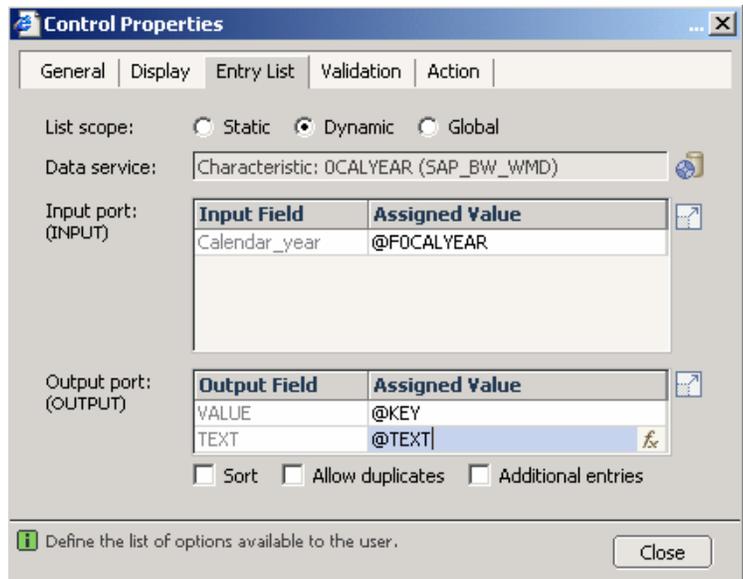
3. Drag and drop a form within your model and name this form "Selection".



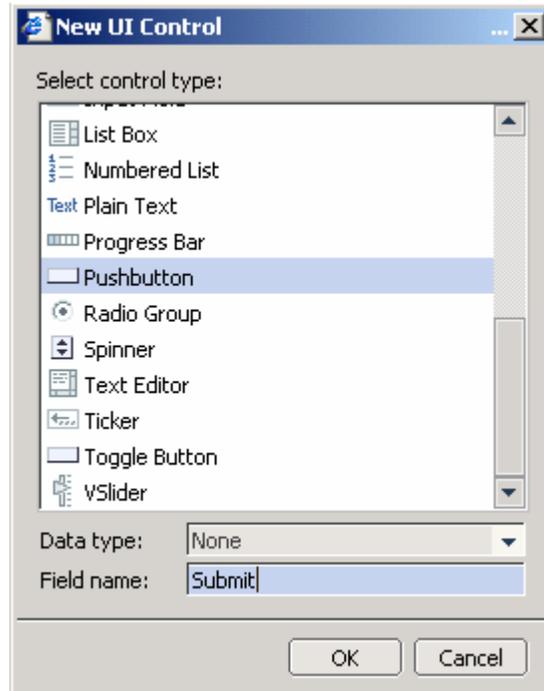
4. Add a dropdown list within your form and name this field "OCALYEAR". Keep in mind that fields cannot start with "0" within VC and the field will automatically be renamed to "FOCALYEAR".



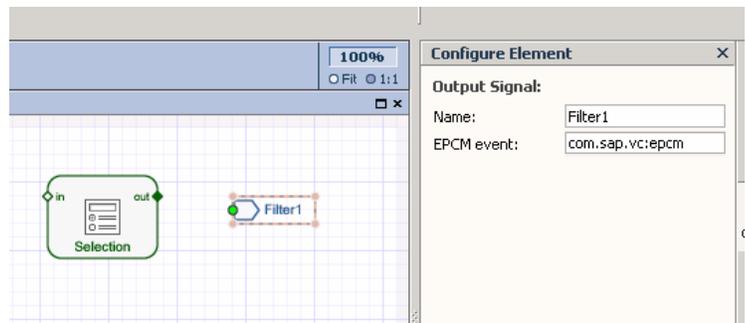
5. Specify either a static or dynamic dropdown list in the properties of this object.



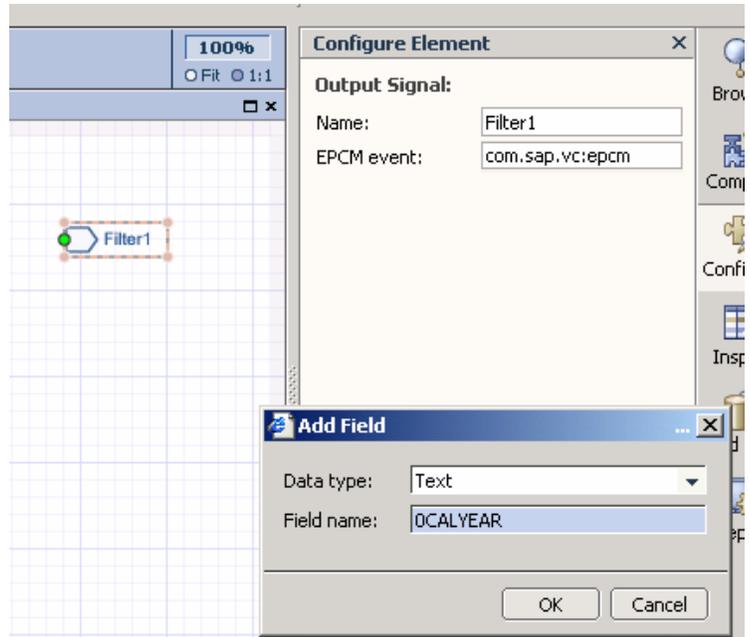
6. Add a Submit button to your form.



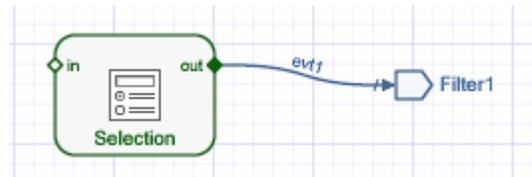
7. Add a Signal out item and make sure the name is "Filter1" and the namespace is specified to "com.sap.vc:epcm". This is very important!!!



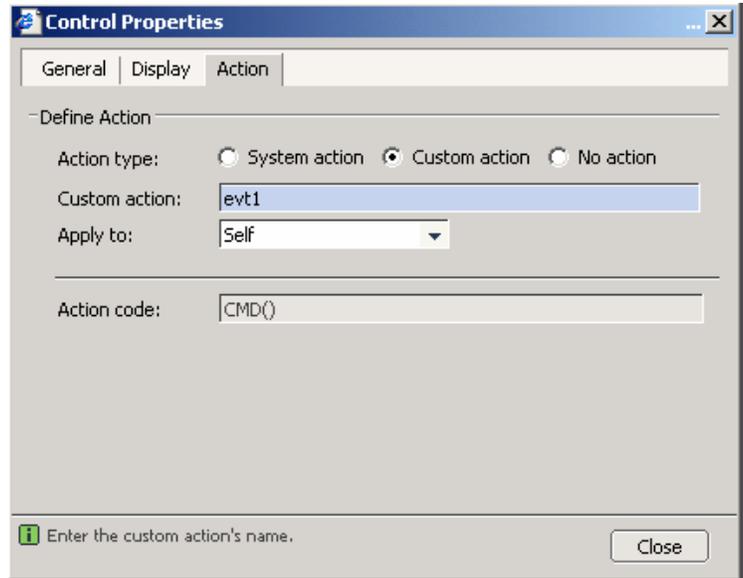
8. Add the field "OCALYEAR" to your signal out.



9. Connect your "Selection" form with the "Filter1" signal out ...



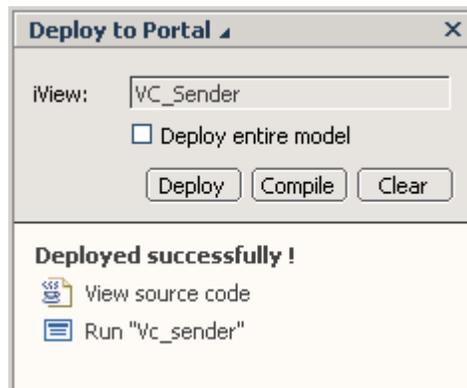
10. Update the properties of the submit button to call custom action "evt1".



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"

```

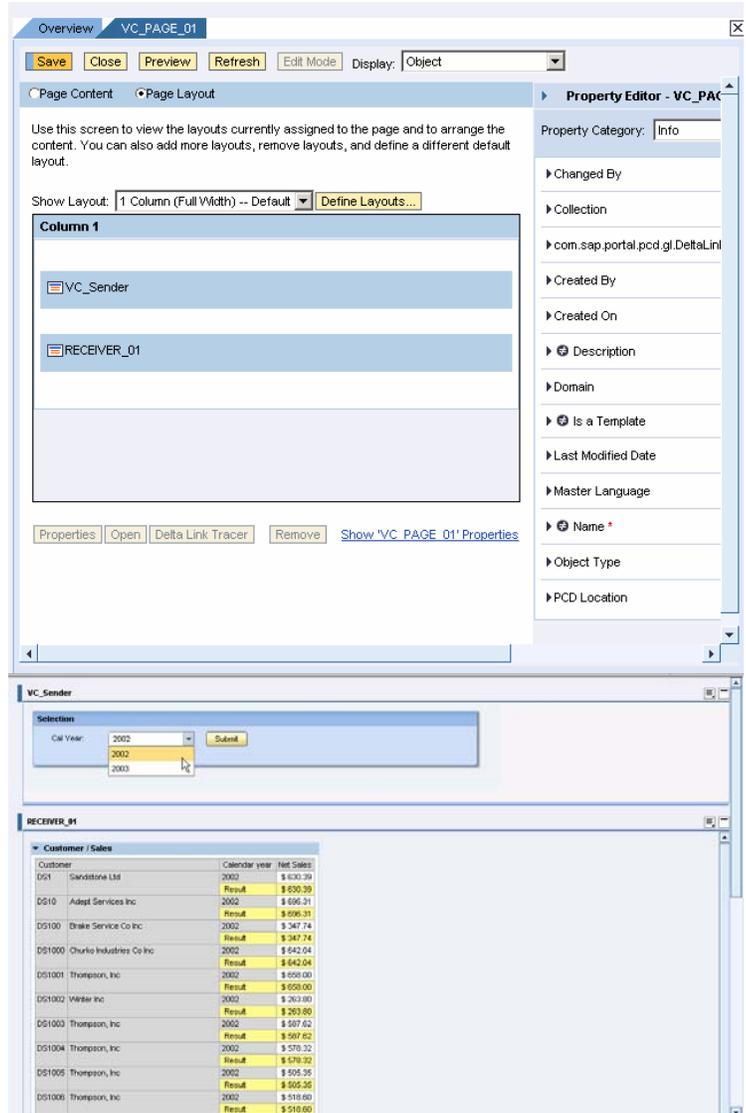
```
value="CL_RSR_WWW_ITEM_GRID"/>
  <param name="DATA_PROVIDER" value="DATAPROVIDER_1"/>
  ITEM:      TABLE_1
</object></P>
</BODY>
</HTML>
```

#### 14. Build an iView for your BI Web Application.

The screenshot displays the 'Property Editor - RECEIVER\_01' window. At the top, there are buttons for 'Save', 'Close', 'Preview', 'Refresh', and 'Edit Mode', along with a 'Display' dropdown set to 'Object'. The 'Property Category' is set to 'Content - BEx Web Application'. The editor contains several expandable sections:

- Application Parameters:** An empty text input field.
- BEx Web Application Query String:** A text input field containing 'TEMPLATE\_ID=ZPD\_RECEIVER\_VC\_01'.
- Customer Exts for 'ParameterProvider':** An empty text input field.
- Don't forward these Parameters to BEx Web Application:** An empty text input field.
- Interval Time to Refresh iView:** A time selection control with fields for hours (0), minutes (0), seconds (0), and milliseconds (-1).
- Logon Language:** A dropdown menu currently showing '- Select -'.
- Parameters forwarded to BEx Web Application:** An empty text input field.
- Show Debug Screen:** Radio buttons for 'Yes' and 'No', with 'No' selected.
- Show Loading Message:** Radio buttons for 'Yes' and 'No', with 'Yes' selected.
- System:** A text input field containing 'SAP\_BW\_VMD'.
- Version:** A dropdown menu showing 'SAP BW 2.x/3.x'.

- 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.



- 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>

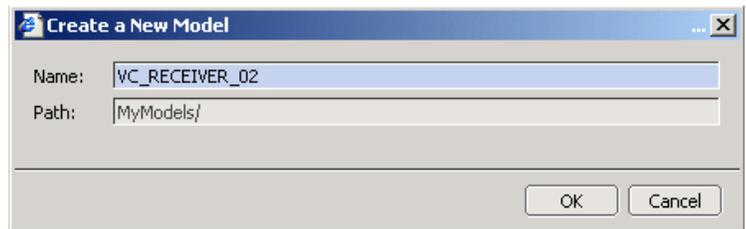
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing
Control">
<TITLE>BW Web Application</TITLE>
  <link href="/sap/bw/Mime/BEx/StyleSheets/BWReports.css"
type="text/css" rel="stylesheet"/>
  <script language=javascript>
<!--
    function raiseEvents(value) {
      extvalue = "<Params 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>

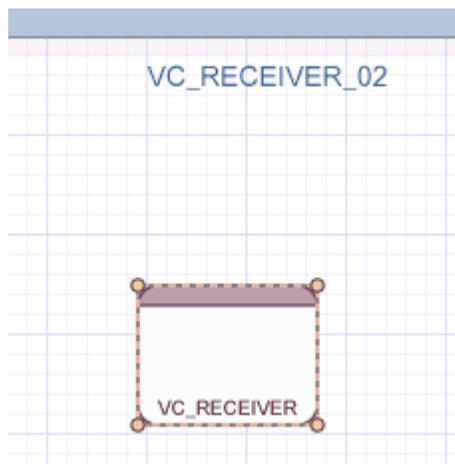
    <TR>
      <TD vAlign=top></TD></TR></TABLE></P>
</BODY>
</HTML>

```

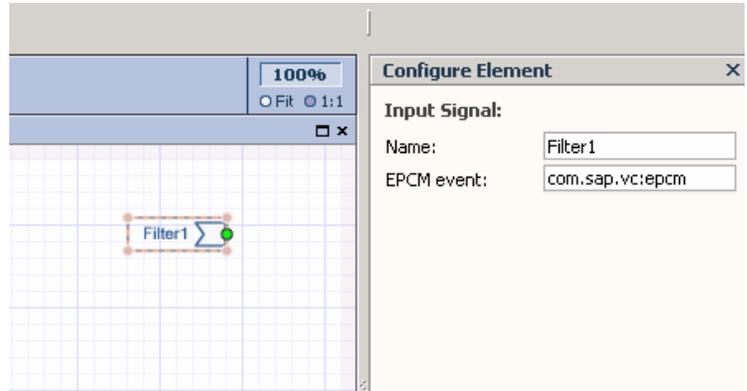
2. Create a new model within VC.



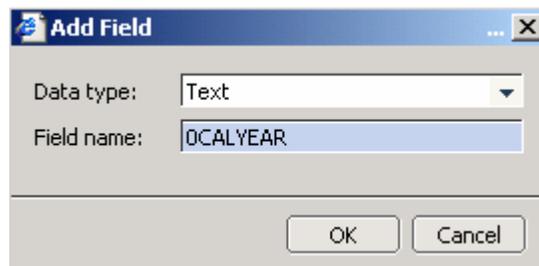
3. Drag and drop an iView within your model and name iView "VC\_RECEIVER".



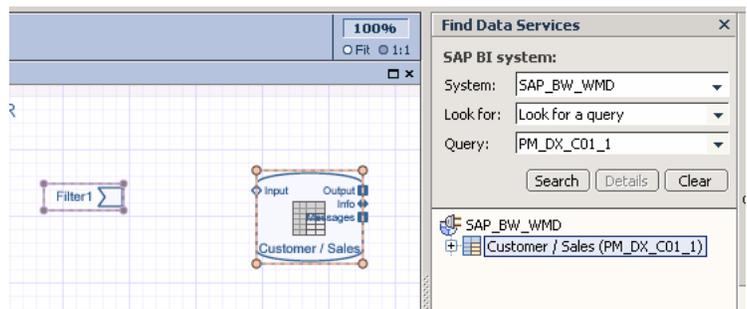
4. Add a signal in called "Filter1" on namepace "com.sap.vc:epcm"



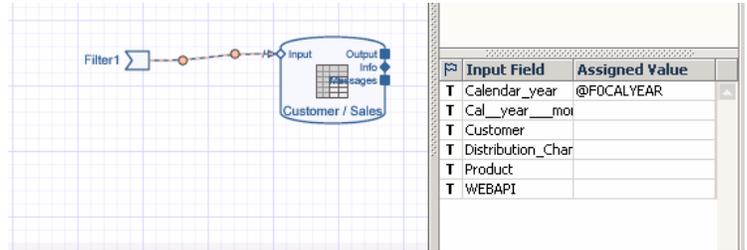
5. Add a field called "OCALYEAR" to your signal in.



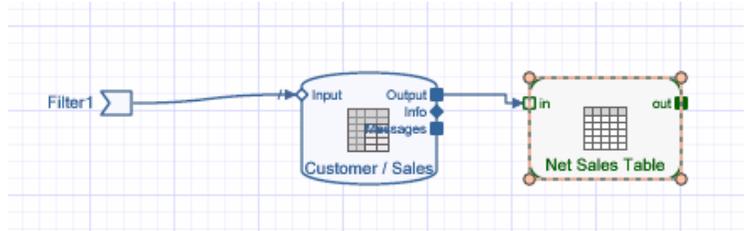
6. Add a data service that you want to pass this filter to ...



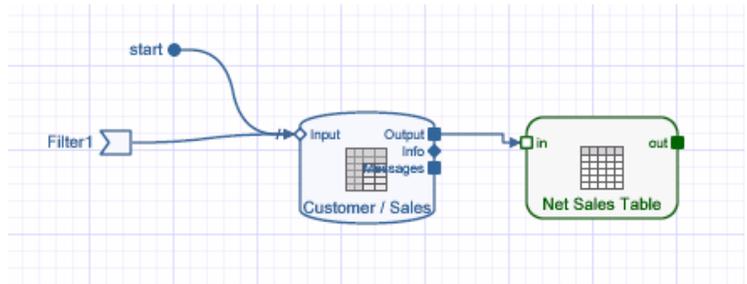
7. Connect the input signal to your data service and specify the mapping for the field you added.



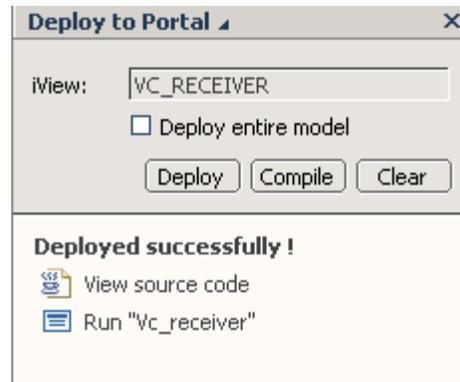
8. Add a table to display the results of the data service and specify your fields within the table.



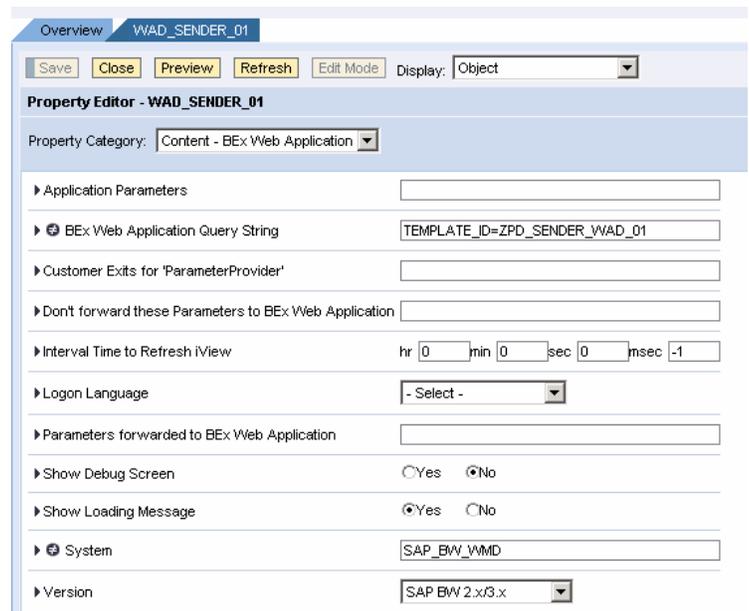
9. Add a start point to your model.



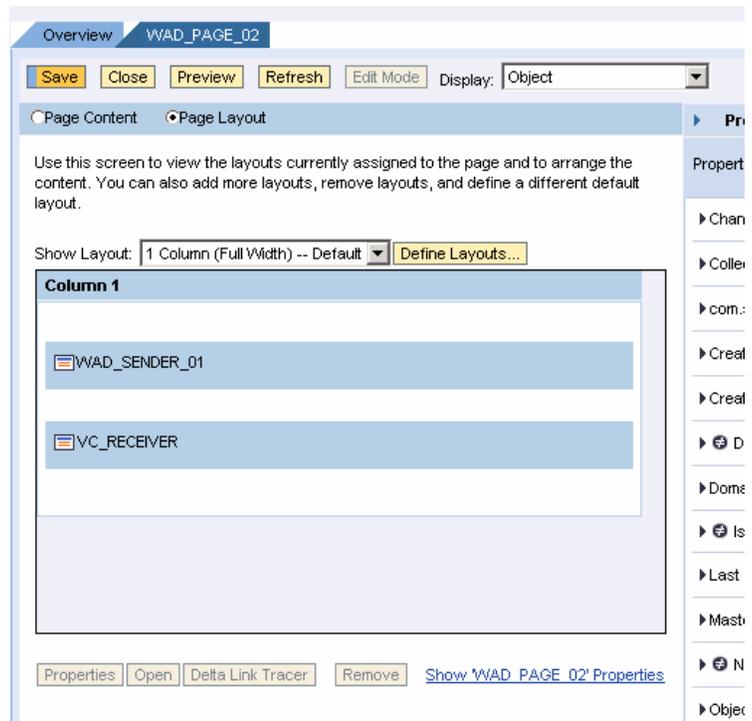
10. Save and deploy your model.



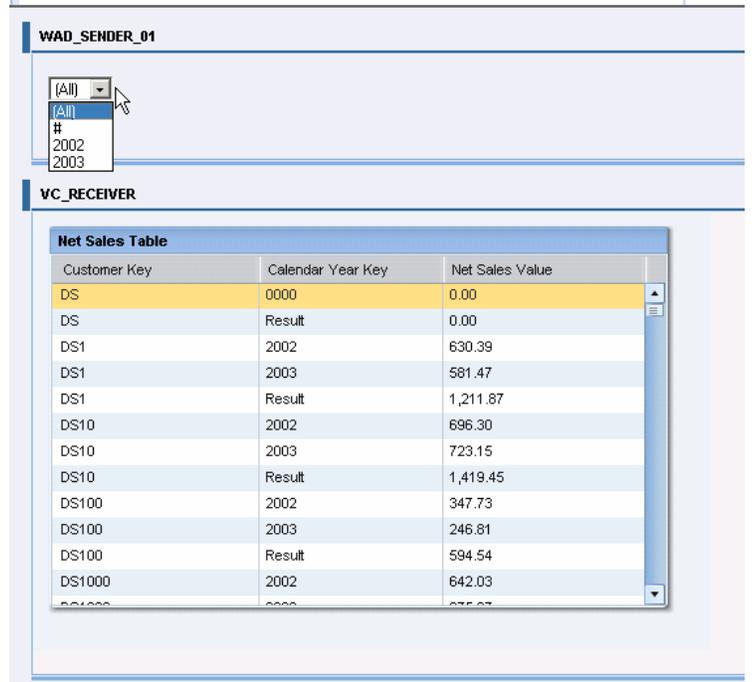
11. Build an iView for your BI Web Application.



- 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.



- 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.

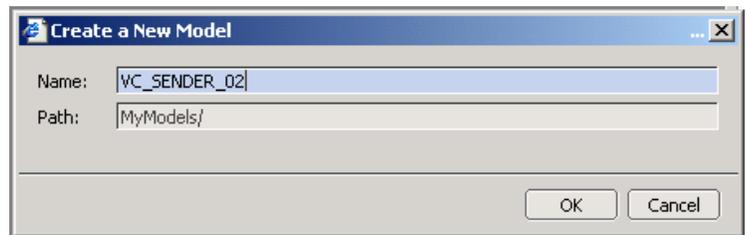


### 4.3 Dynamic Events from Visual Composer to Web Application Designer

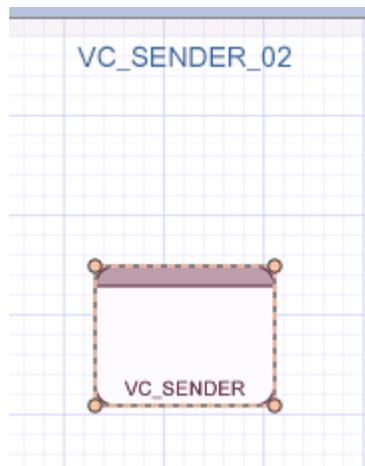
This solution using javascript within the Web Application Designer to read all filter values passed from Visual Composer and dynamically builds the web api string for filtering. This requires your Visual Composer to name your objects on your signal out with the technical name of the

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

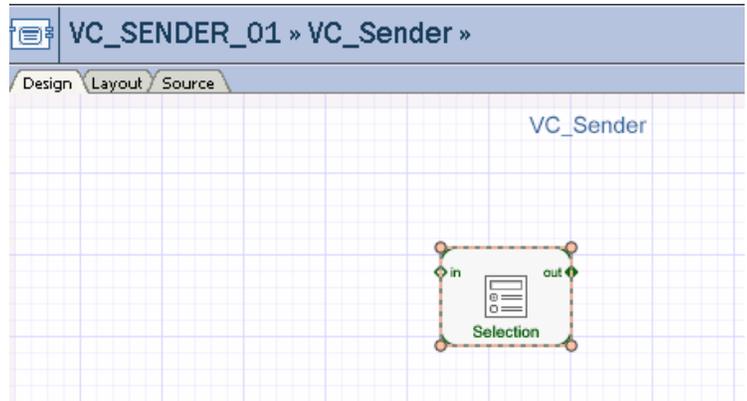
1. Create a New Model within VC.



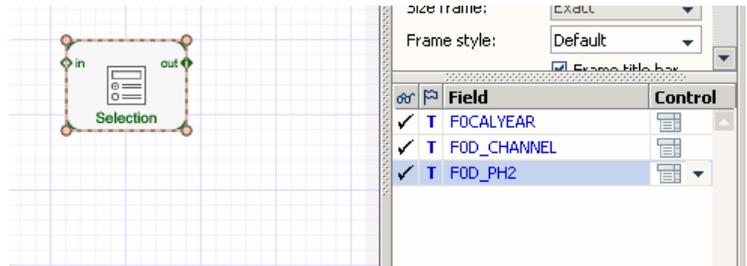
2. Drag and drop an iView within your model and name iView "VC\_SENDER".



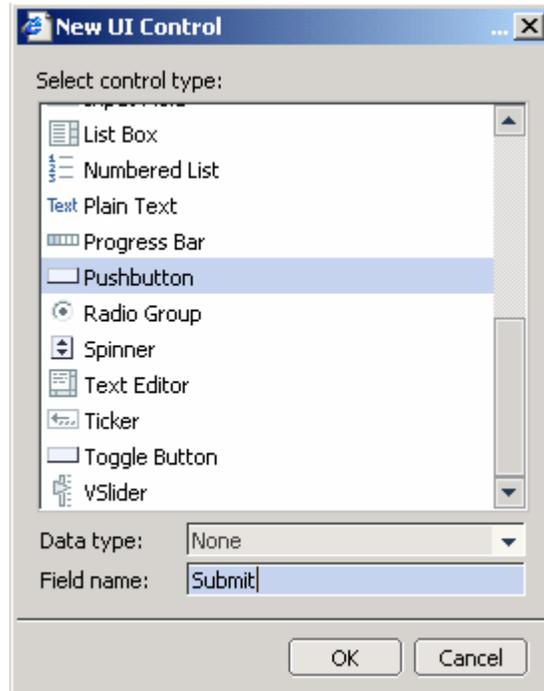
3. Drag and drop a form within your model and name this form "Selection".



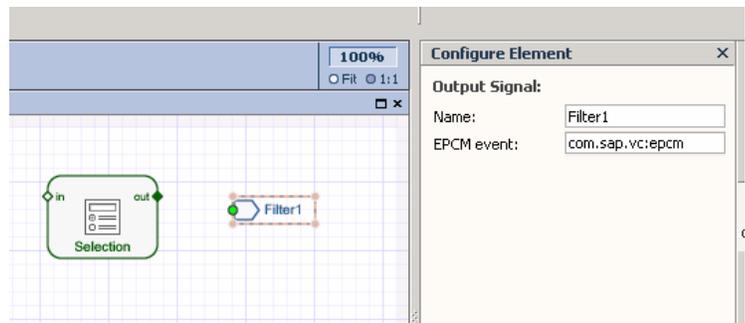
4. Add 3 fields that are defined off of dropdown lists based off either static or dynamic entry lists (could be based off characteristic).



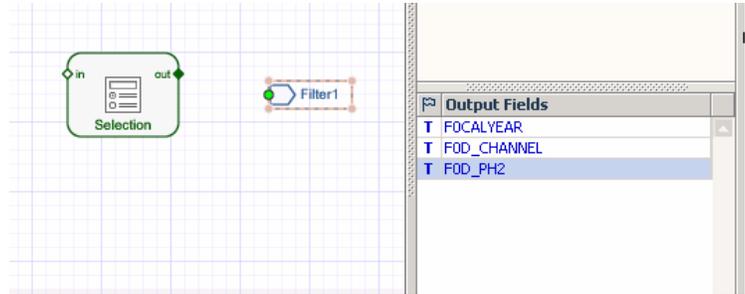
5. Add a Submit button to your form.



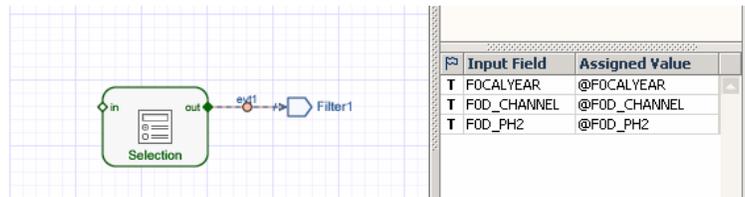
6. Add a Signal out item and make sure the name is "Filter1" and the namespace is specified to "com.sap.vc:epcm". This is very important!!!



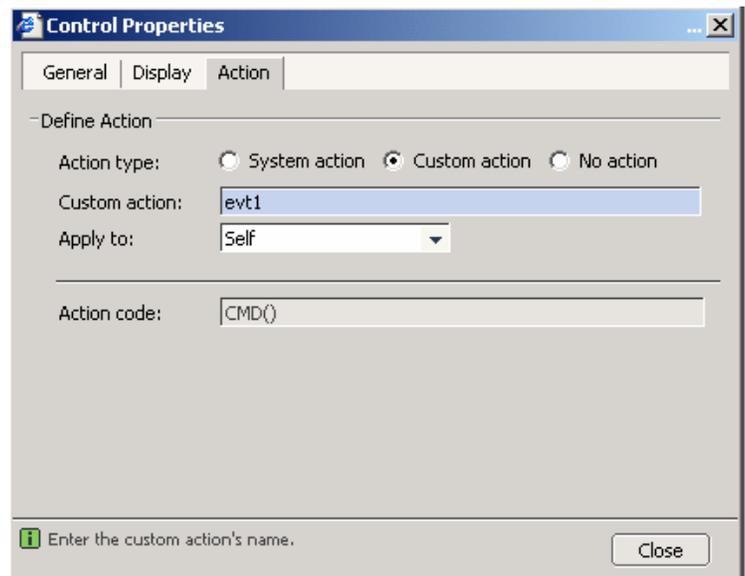
7. Add your 3 fields to the output signal. Make sure these fields are named after the technical name of an infoobject. Also, keep in mind that objects that start with zero automatically have the character "F" added to the front.



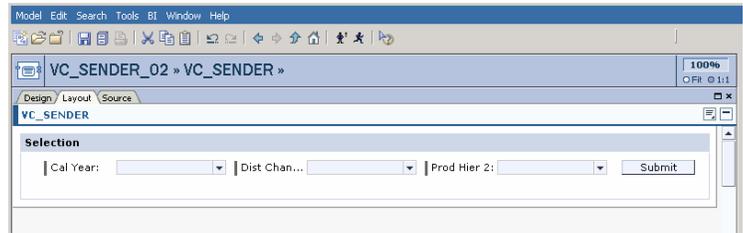
8. Connect your "Selection" form with the "Filter1" signal out ... Ensure that the mapping is done.



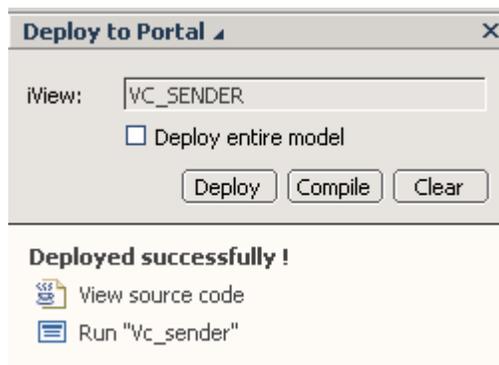
9. Update the properties of the submit button to call custom action "evt1".



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");

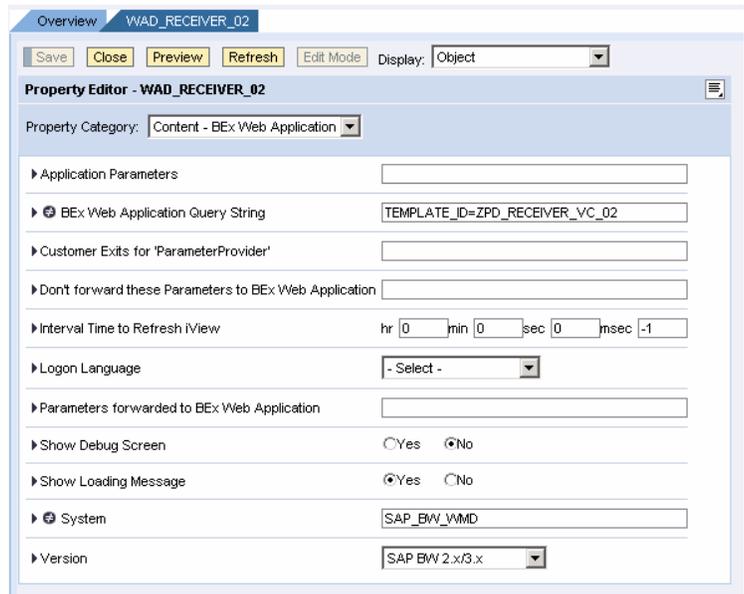
function myreceiveEvent( eventObj )
{
  values = eventObj.dataObject;
  unesc_val = unescape(values);
  load(unesc_val);
  len_val =
xmlDoc.getElementsByTagName("Row").item(0).attributes.length;
  var valueArray = new Array();
  valueArray =
xmlDoc.getElementsByTagName("Row").item(0).attributes;
  var 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>

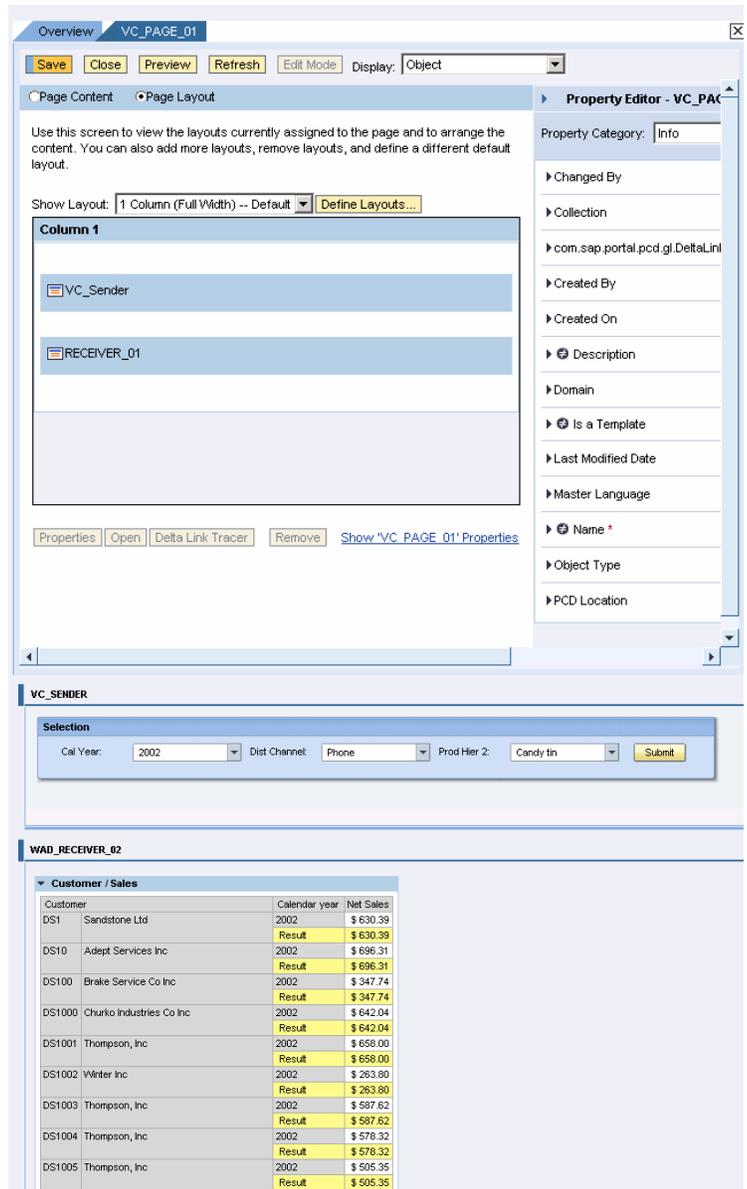
</HEAD>
<BODY>
<P><object>
  <param name="OWNER" value="SAP_BW"/>
  <param name="CMD" value="GET_ITEM"/>
  <param name="NAME" value="TABLE_1"/>
  <param name="ITEM_CLASS" value="TABLE_1"/>
</P>
</BODY>
</HTML>
```

```
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.



- 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.



- 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>
```

- 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.

```

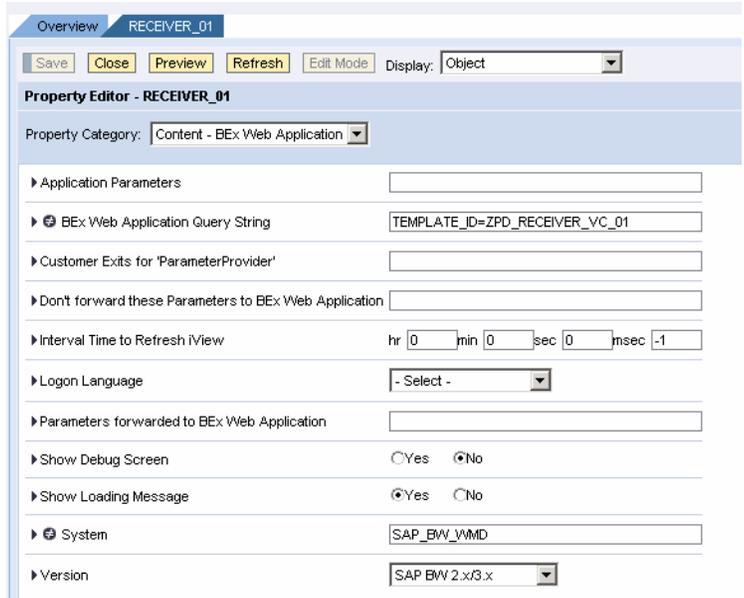
<!-- BW data source object tags -->
<object>
  <param name="OWNER" value="SAP_BW"/>
  <param name="CMD" value="SET_DATA_PROVIDER"/>
  <param name="NAME" value="DATAPROVIDER_1"/>
  <param name="QUERY" value="PM_DX_C01_1"/>
  <param name="INFOCUBE" value="0D_DX_C01"/>
  DATA_PROVIDER:      DATAPROVIDER_1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

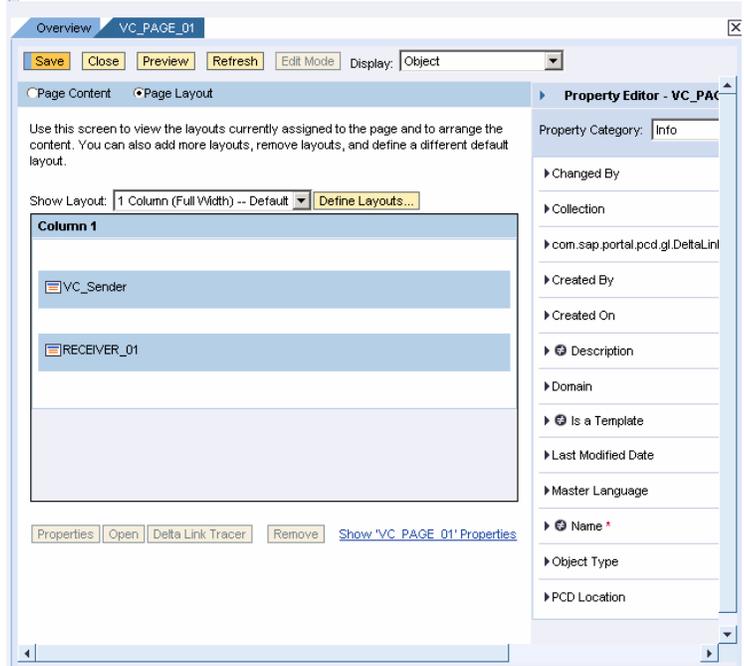
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing
Control">
<TITLE>BW Web Application</TITLE>
  <link href="/sap/bw/Mime/BEx/StyleSheets/BWReports.css"
type="text/css" rel="stylesheet"/>
  <script language=javascript>
EPCMPROXY.subscribeEvent( "urn:com.sap.vc:epcm", "Filter1",
window, "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.



- 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!!!

WAD\_SENDER\_01

[All] # 2002 2003

WAD\_RECEIVER\_02

Customer / Sales

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!!!

- Create a New Model within VC.

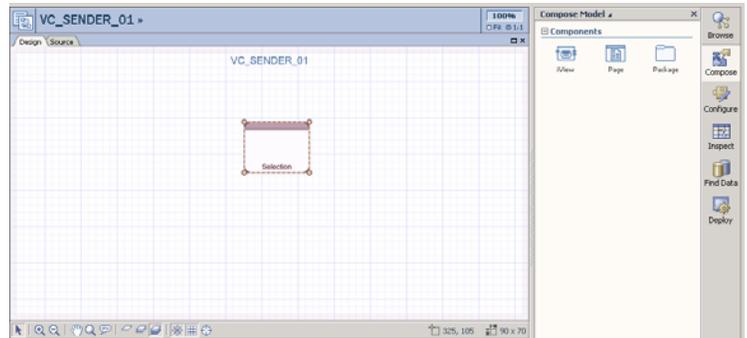
Create a New Model

Name: WC\_SENDER\_01

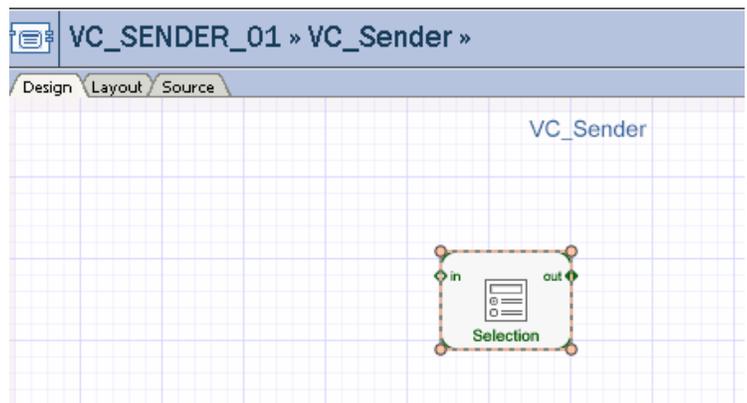
Path: MyModels/

OK Cancel

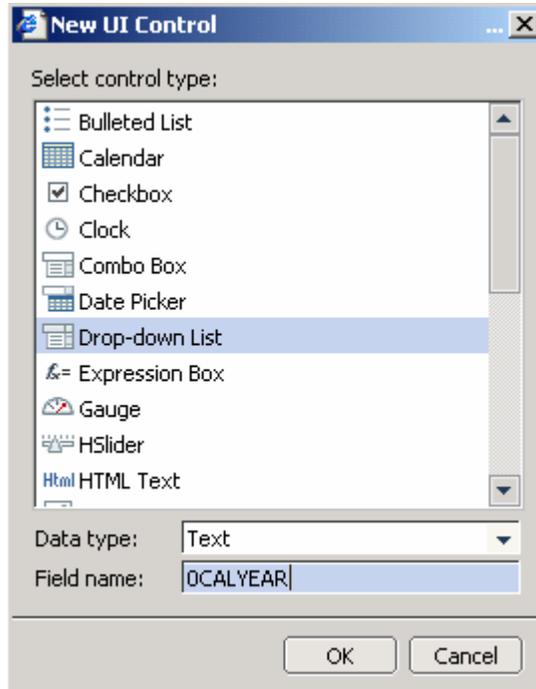
2. Drag and drop an iView within your model and name iView "VC\_Sender".



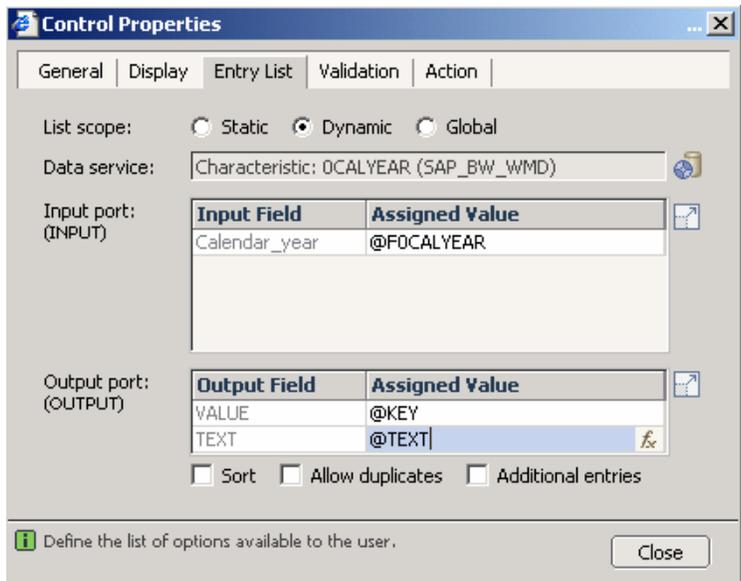
3. Drag and drop a form within your model and name this form "Selection".



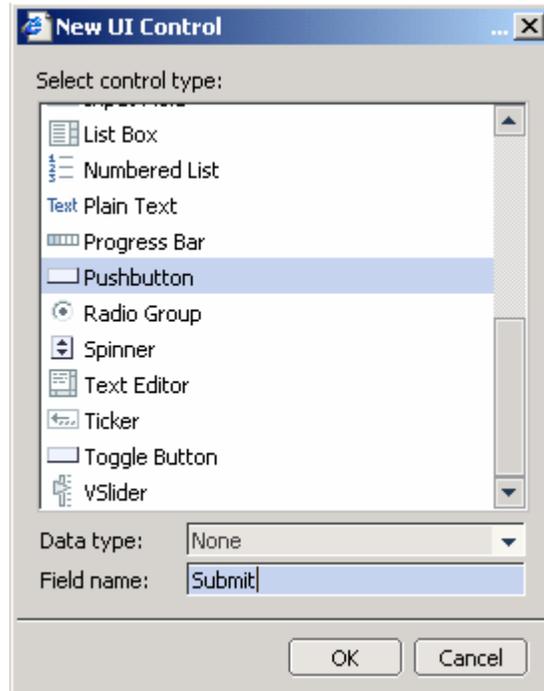
4. Add a dropdown list within your form and name this field "OCALYEAR". Keep in mind that fields cannot start with "0" within VC and the field will automatically be renamed to "FOCALYEAR".



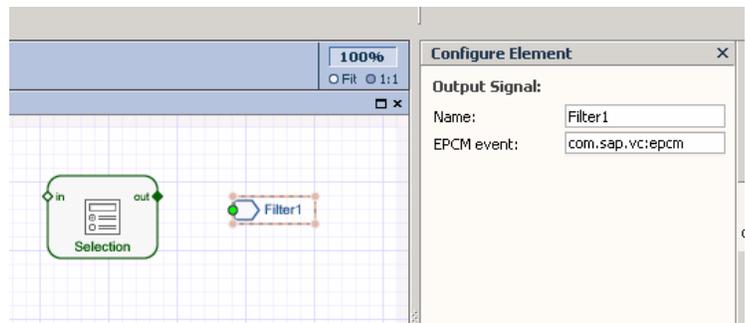
5. Specify either a static or dynamic dropdown list in the properties of this object.



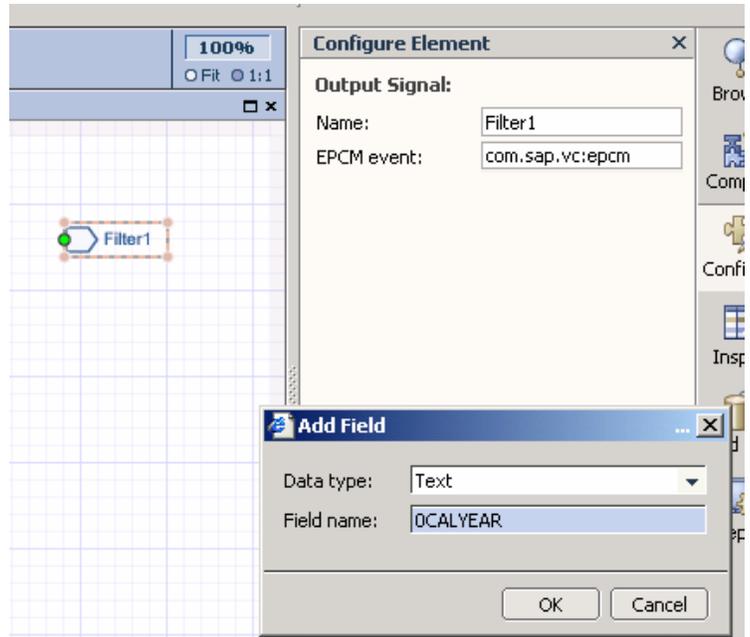
6. Add a Submit button to your form.



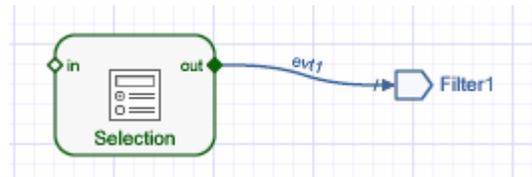
7. Add a Signal out item and make sure the name is "Filter1" and the namespace is specified to "com.sap.vc:epcm". This is very important!!!



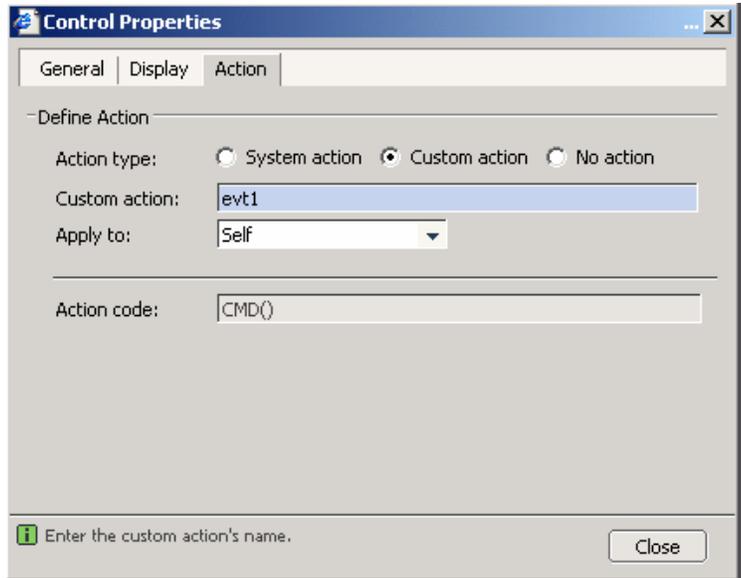
8. Add the field "OCALYEAR" to your signal out.



9. Connect your "Selection" form with the "Filter1" signal out ...



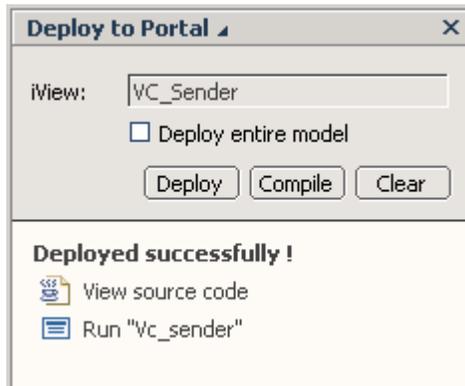
10. Update the properties of the submit button to call custom action "evt1".



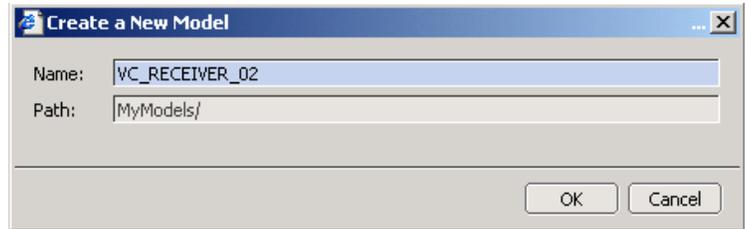
11. Adjust the layout of your VC Model.



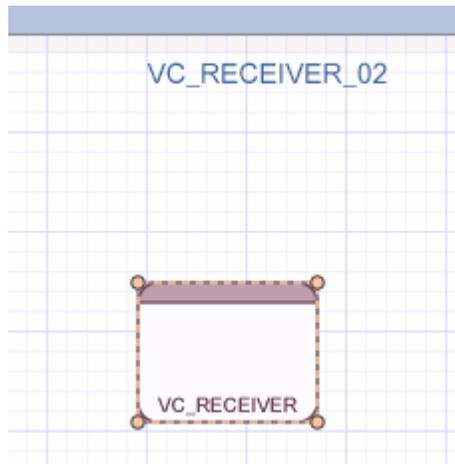
12. Save and Deploy your model.



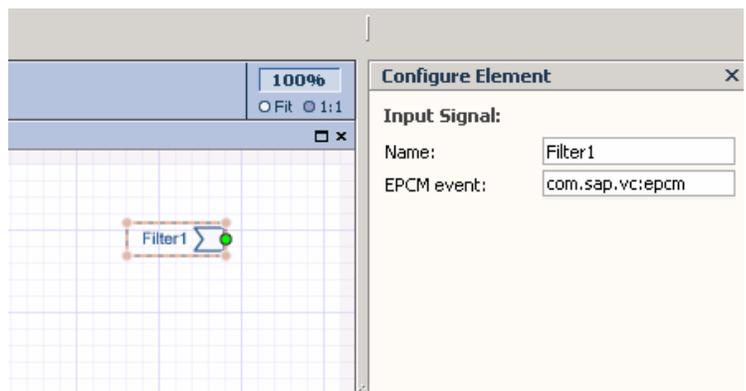
13. Create a new model within VC.



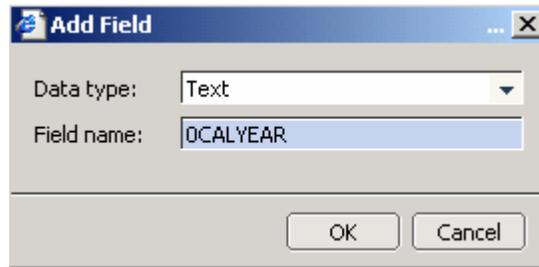
14. Drag and drop an iView within your model and name iView "VC\_RECEIVER".



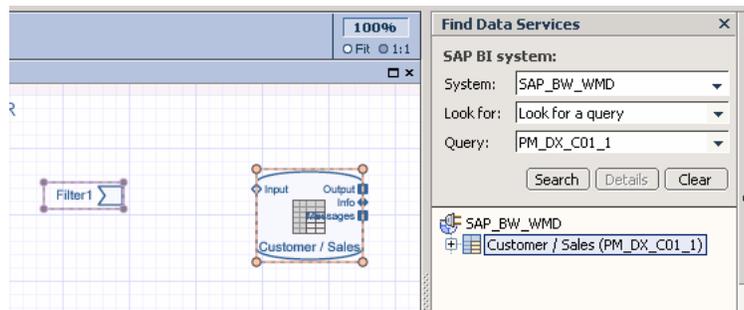
15. Add a signal in called "Filter1" on namespace "com.sap.vc:epcm"



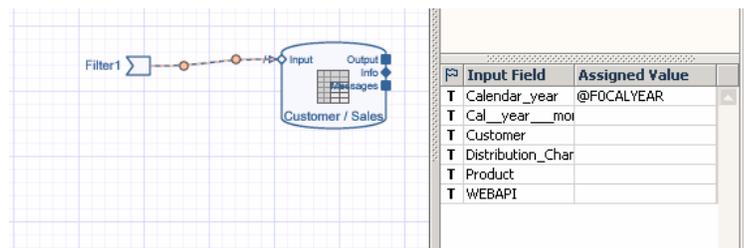
16. Add a field called "OCALYEAR" to your signal in.



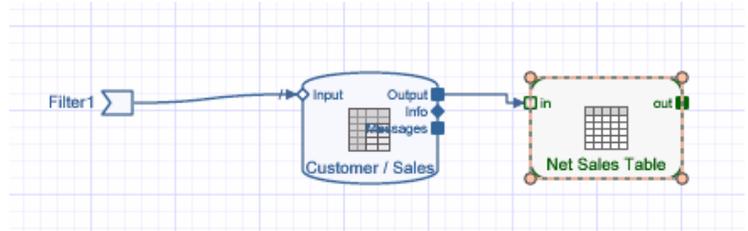
17. Add a data service that you want to pass this filter to ...



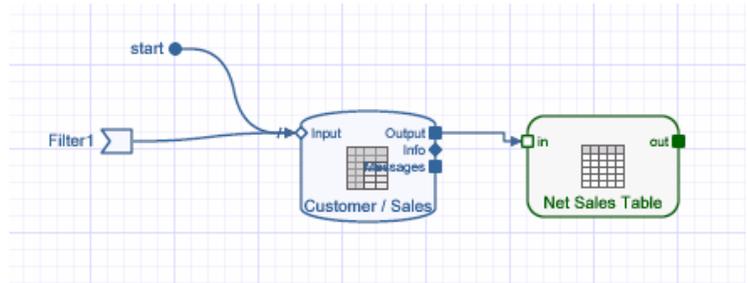
18. Connect the input signal to your data service and specify the mapping for the field you added.



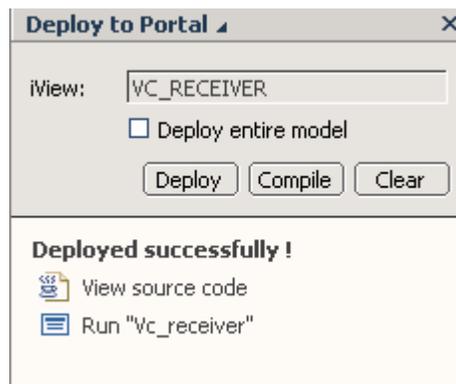
19. Add a table to display the results of the data service and specify your fields within the table.



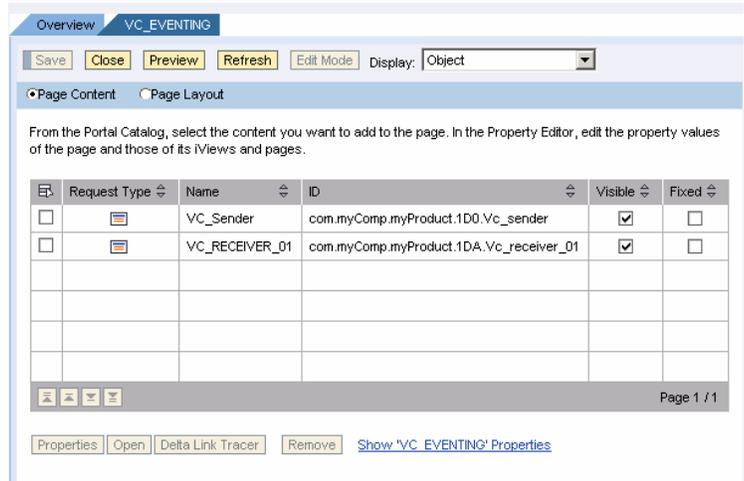
20. Add a start point to your model.



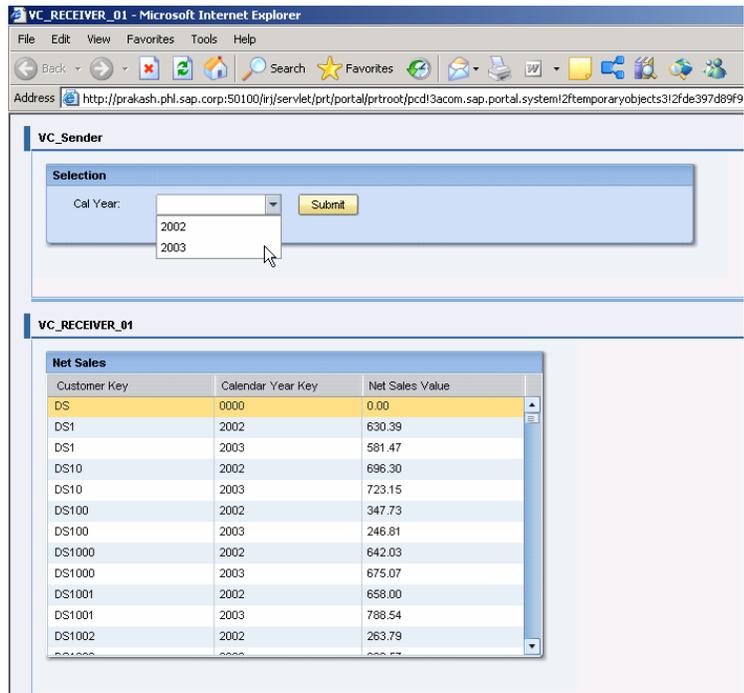
21. Save and deploy your model.



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



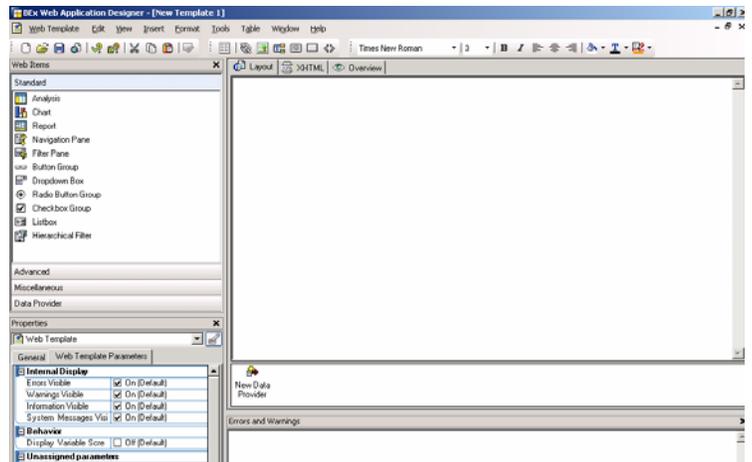
23. Run your Portal Page and see the modular components!!!



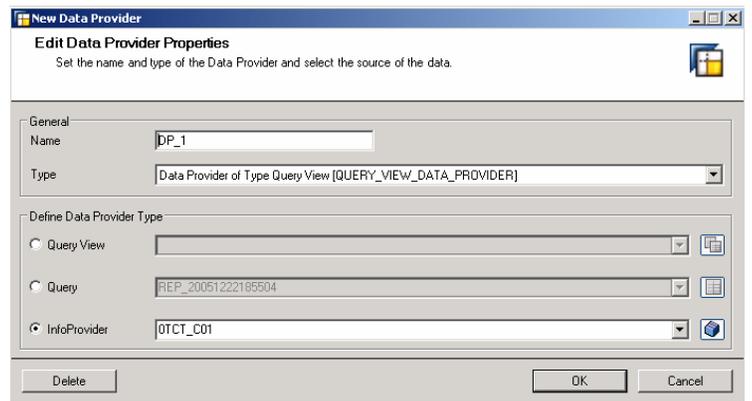
#### 4.6 Working with Javascript within the SAP NetWeaver 2004s Web Application Designer

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.

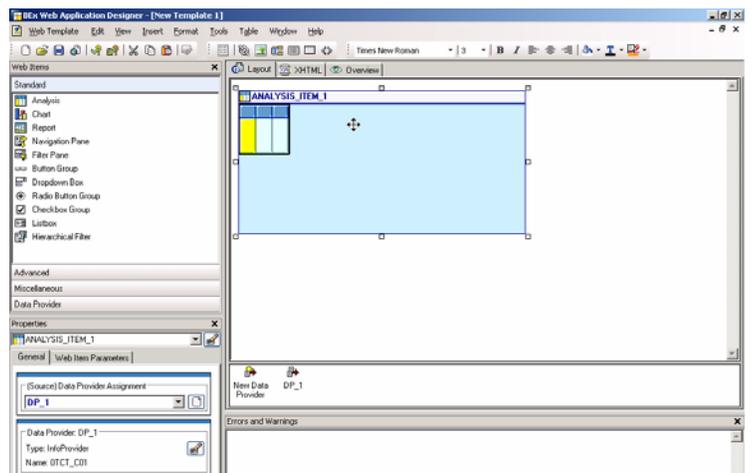
1. Launch the new web application designer.



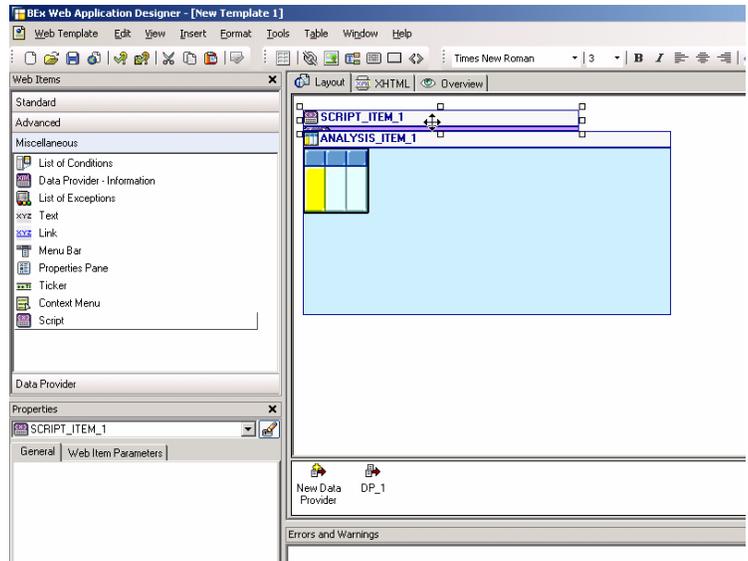
2. Create a new DataProvider.



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

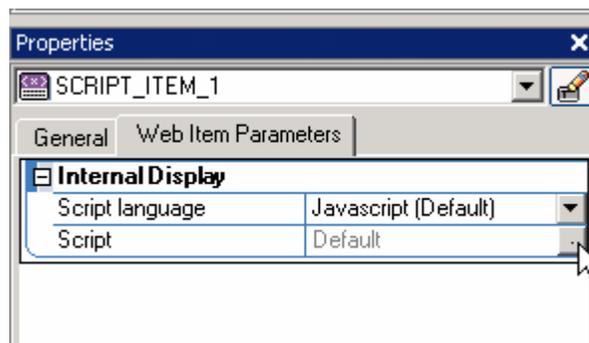


4. Drag and drop the "Script" item into your web template.

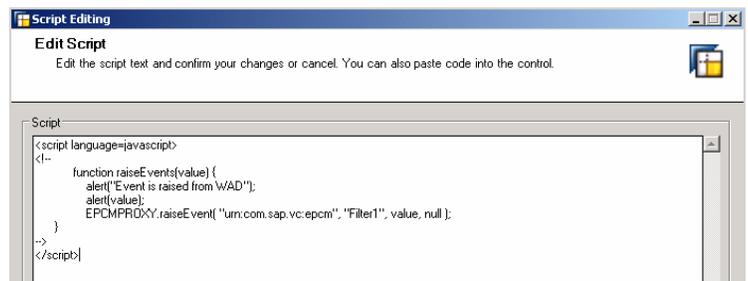


5. Within the web item properties, choose the button to enter your javascript code.

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.

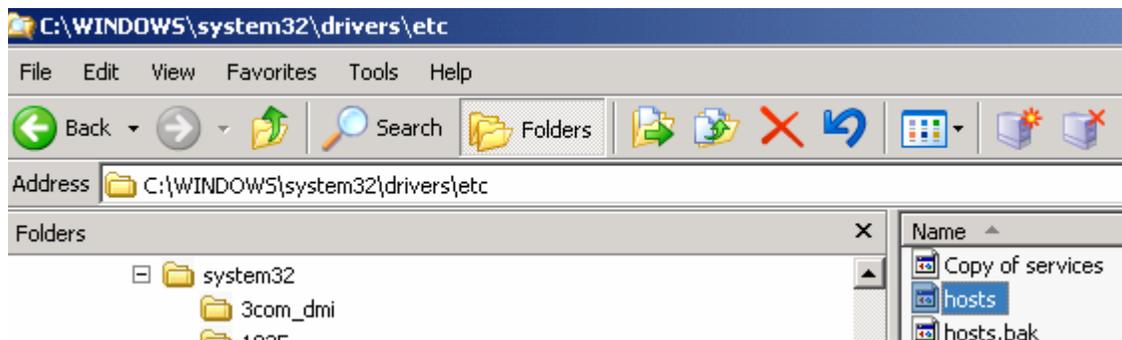


## 5 Common Issues and Resolutions

### 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
```

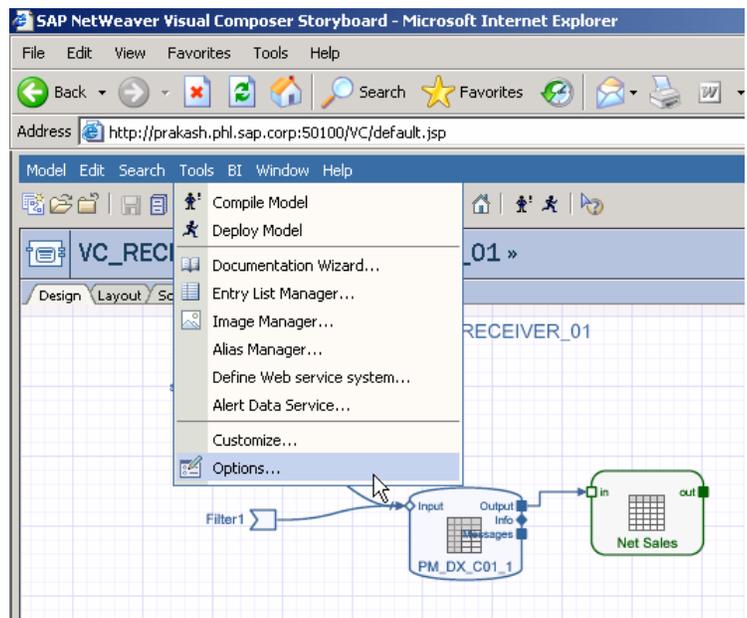
Because this is client side eventing, this hosts file entry would need to be specified on the CLIENT side. If it's on the server you would need to register a fully qualified domain name in the same domain for both systems and have users access these url's from the same domain.

# 6 Troubleshooting

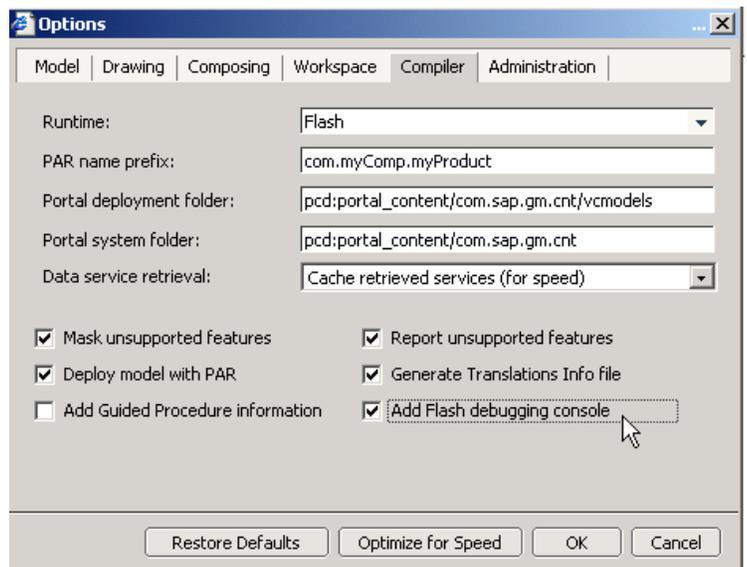
## 6.1 Flash Debugger

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.

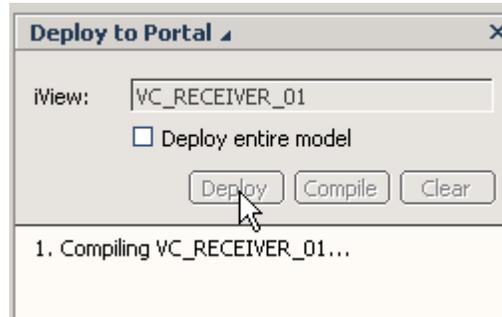
1. Within Visual Composer, go to Tools -> Options.



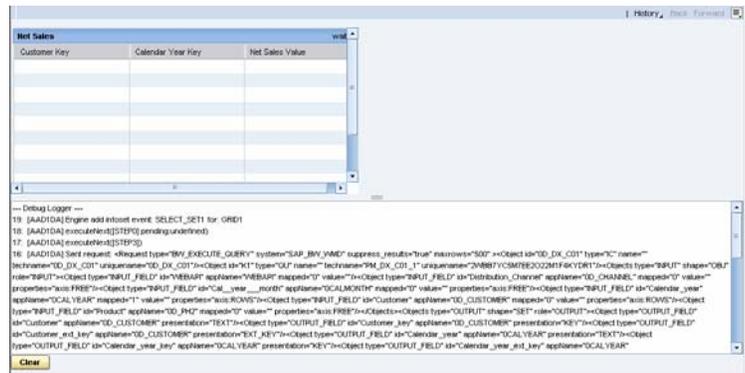
2. Go to the Compiler tab. Choose the "Add Flash debugging console" option and choose OK.



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".

The screenshot shows the SAP Enterprise Portal interface. At the top, there is a navigation bar with the following items: Home, Home, Content Administration, User Administration, System Administration, and E. Below this, there is a search bar and a sub-navigation bar with items: Transport, Monitoring, Permissions, System Configuration, Portal Display, and Support. The main content area is titled 'Support Desk' and contains a 'Detailed Navigation' section with a tree view showing 'Support Desk' and 'Web Dynpro Test Tools'. To the right of the navigation is a text block: 'This Support Desk is designed to help you perform main portal areas with links to their specific support you can perform automatic configuration checks to information and test tools with each support packag'. Below the navigation is a 'Portal Favorites' section. On the right side, there is a 'Top Level Areas' section with a table:

Area	Description
<a href="#">Client Framework</a>	Sharing data on the client
<a href="#">Navigation</a>	Top-level navigation, details
<a href="#">Portal Content Directory</a>	Repository for roles, work

- Choose Eventing to register your events.

Info	Description
Central Note	892359
Component	EP-PIN-CS

**Major Known Issues and Solutions**

Issue	Solution
Browser prerequisites	JavaScript and cookies must be enabled in user's browser.

**Test and Configuration Tools**

Tool	Description
<a href="#">com.sap.portal.epcf.loader.Environment</a>	Test the system functions, especially user agent detection.
<a href="#">com.sap.portal.epcf.loader.Eventing</a>	Test the eventing, provides Configurable event sender receiver.
<a href="#">com.sap.portal.epcf.loader.Eventing.unsubscribe</a>	Test the subscribing and unsubscribing client events.
<a href="#">com.sap.portal.epcf.loader.DataBag</a>	Test the client databag.
<a href="#">com.sap.portal.epcf.loader.Dirty</a>	Test the Cross Navigation and WorkProtect feature.
<a href="#">com.sap.portal.epcf.loader.Relaxing</a>	Test of Relaxing of document domain (JavaScript origin policy)

- Enter your namespace and event and attach event.

urn:	<input type="text" value="urn:com.sap.vc:epcm"/>
name:	<input type="text" value="Filter1"/>
Event Log:	<div style="border: 1px solid gray; padding: 5px;"> 14:10:31 : urn:com.sapportals.portal:browser, load, [object], undefined  14:10:31 : urn:com.sapportals.portal:browser, preload, [object], undefined </div>
<input type="button" value="Attach Event"/> <input type="button" value="Show Attached Events"/> <input type="button" value="Clear Event Log"/>	

- 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.

**Client Framework: Eventing**

Presets:	urn:com.sapportals.portal:EpcTest, A	Store Preset
urn:	urn:com.sap.vc:epcm	
name:	Filter1	
dataobject:	2002	
sourceid:		
Raise Event		

## 6.4 HttpWatch

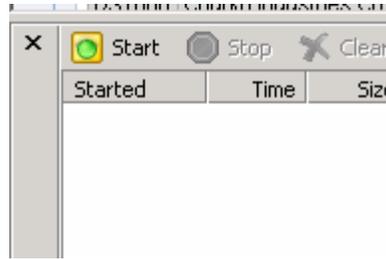
HttpWatch allows you to track html calls. This requires that this software is installed. Also, keep in mind that this doesn't offer you internal commands to flash and will help with eventing between portal iViews, but not off much assistance within the flash runtime. The Flash Debugger should be used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

The screenshot shows the SAP NetWeaver Portal interface in Microsoft Internet Explorer. The address bar shows a URL with a query parameter for HTTPWATCH. Below the browser window, there are two toolbars: WAD\_SENDER\_01 and WAD\_RECEIVER\_02. The WAD\_RECEIVER\_02 toolbar displays a table with the following data:

Customer / Sales			
Customer	Calendar year	Net Sales	
DG	DG	#	\$ 0.00
		Result	\$ 0.00
DG1	Sandstone Ltd	2002	\$ 630.39
		2003	\$ 501.40
		Result	\$ 1,211.89
DG10	Adept Services Inc	2002	\$ 686.21
		2003	\$ 723.15
		Result	\$ 1,419.46
DG100	Drake Service Co Inc	2002	\$ 347.74
		2003	\$ 248.81
		Result	\$ 594.55

2. Choose Start to log http calls.



3. Trigger an event that will update other iViews.

A screenshot of a Microsoft Internet Explorer browser window displaying the SAP NetWeaver Portal. The address bar shows the URL: `http://prakash.phl.sap.corp:50100/irj/servlet/prt/portal/prtroot/pcd13aportal_content!2f8`. The page content includes a section titled 'WAD\_SENDER\_01' with a dropdown menu showing options: '(All)', '#', '2002', and '2003'. Below this is a section titled 'WAD\_RECEIVER\_02' containing a table with the following data:

Customer / Sales			
Customer		Calendar year	Net Sales
DS	DS	#	\$ 0.00
		Result	\$ 0.00
DS1	Sandstone Ltd	2002	\$ 630.39
		2003	\$ 581.48

At the bottom of the browser window, a log window is open, showing a table of HTTP calls:

Started	Time	Size	Op.	Result	Type	URL
00:00:00.000	0.300	8981	GET	200	text/html; char...	http://rgph101.phl.sap.corp:1080/sap
00:00:00.501	0.000	0	GET	(Cache)	text/javascript	http://rgph101.phl.sap.corp:1080/sap
00:00:00.541	0.000	0	GET	(Cache)	text/javascript	http://rgph101.phl.sap.corp:1080/sap
00:00:00.561	0.060	0	GET	(Cache)	text/html	http://rgph101.phl.sap.corp:1080/sap

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%22%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%22%22%20%3E%3CRow%20F0FISCPER%3D%222002%22%20F0CALYEAR%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](http://www.sap.com/netweaver)