EDM EMM Scenario Part III: UI Modeling with SAP NetWeaver Visual Composer

Applies to:
SAP NetWeaver 2004s SP10 – IT Scenario “Developing, Configuring, and Adapting Applications”, IT Scenario Variant “Creating Composite Applications.”
Frontend:
Microsoft Internet Explorer 6.0 SP1 or higher
Microsoft XML Parser 4.0
Adobe SVG Viewer 3.0

Summary
This tutorial is a part III of a series that describes core concept of composite applications and how to build them with NW04s based upon the EDM demo scenario as been shown on several events by the Market Development Engineering team.
The focus of this document lies on the UI modelling using the Visual Composer (VC) a graphical design tool to compose portal content.

Author(s): Mark Mauerwerk
Company: Axentiv AG
Created on: 11th January 2007

Author Bio
Mark Mauerwerk is currently working as Solution Architect of SAP Platform Ecosystem primarily focused on developing early stage demos and technical due diligence of SAP’s newest technologies. He is an expert in SAP NetWeaver architecture (ABAP and JAVA) and senior consultant at axentiv AG, a German SAP service partner and ISV enabling partner. He looks back on many years of experience in development and consultancy both internally at SAP as well as for customers & partners in the field.
Table of Contents

Introduction ...................................................................................................................................... 3
Introduction to Visual Composer (VC) ......................................................................................... 3
Authorization ................................................................................................................................4
Configure Web Service Proxy...................................................................................................... 4
Implementation ................................................................................................................................6
Create a VC Model....................................................................................................................... 6
Create the iViews and Layers...................................................................................................... 6
Model the Presentation Logic ...................................................................................................... 9
Data Sources ................................................................................................................................ 9
Modeling UI Elements and Event Handling............................................................................ 10
Configuration .......................................................................................................................... 11
Customize the Layout ................................................................................................................ 15
Deployment ................................................................................................................................1 5
Testing and Error Analysis............................................................................................................. 16
Testing........................................................................................................................................ 16
Debugging.................................................................................................................................. 16
Related Content............................................................................................................................. 17
Copyright........................................................................................................................................ 18
Introduction

This document is part three of the implementation how-to’s for the EDM scenario. It describes the UI modeling with visual composer (VC) for the EDM scenario.

Prerequisite for this part is that you have already implemented and published the composite application services according part II of the EDM How-To guides.

The UI design takes place in the Visual Composer Storyboard a visual design tool that helps you to create portal iViews without writing any line of code.

Within the context of our EDM scenario you will learn how-to:

- Create a VC model
- create pages and iViews accessing web services as data source
- define the relationships between UI controls and data sources
- customize the iView layout
- deploy iViews to portal

Introduction to Visual Composer (VC)

Visual Composer operates on top of the portal, utilizing the portal’s connector-framework interfaces. It allows users to access a range of data sources, including SAP and third-party enterprise systems.

With the Visual Composer client – the Storyboard – you can design all aspects of your iViews including

The Data Service Layer – Basically VC models can be built around any number of disparate data services (including SAP RFCs and BAPIs, BI queries, JDBC-compliant stored procedures, Web services etc.) and then finely tuned to display only the exact subset of data needed for the specific transaction or function

The User Interface (UI) Logic - The UI logic defines interaction between UI controls and the underlying data queries. The story board supports designing the UI by dragging and dropping controls from a toolset panel,
such as the forms it displays at runtime. As you construct the UI logic Visual Composer writes the underlying code.

The UI Layout - The layout defines the visual aspects of the iView. The Storyboard provides WYSIWYG tools for customizing the iView layout.

Because Visual Composer is a graphical design tool, it is suitable to include business users (content experts) during design time to customize pages and iViews as needed.

The models that you build in Visual Composer are generated in a proprietary meta language the Generic Modeling Language (GML). To deploy your application to a portal, the GML code must be compiled into a language supported by the portal. Currently Visual Composer produces iViews based on FLASH and Web Dynpro (limited).

The GML code can be exchanged between different systems. For specifying dynamic behavior GML supports expressions to apply user defined functions at runtime.

As you build your model in Storyboard, you drill down from upper-level model entities down to the lowest-level elements and controls.

**Authorization**

The required authorizations for VC are assigned to the role pcd:portal_content/com.sap.gm.cnt/core/VCRole.

**Configure Web Service Proxy**

SAP provides the capability to expose all web services on NetWeaver through the Web Service Inspection Language (WSIL). WSIL is an XML document format to facilitate the discovery and aggregation of Web Service descriptions in a simple and extensible fashion (you can display your wsil document on http://<server>:<port>/inspection.wsil). Once you have configured a WSProxy client against your system providing the web services you do not need to define a Web service system for each web service in VC.

In order to connect your J2EE and also your Visual Composer to your system exposing your WS you need to configure two connections within the Visual Administrator of the J2EE engine.

Start out by connecting to Visual Administrator with an administrative user.

Navigate to the "Web Services Security" service under the Server node and then open up "Web Service Clients" -> "sap.com" and highlight “DynamicWSProxies” and create a new entry.
In the popup appearing after you have clicked the “Create” button and specify the name of your system.

Now the connection is displayed in the tree under DynamicWSProxies. Click on the newly created Connection and fill the connection parameters as can be seen in the above screenshot.

Depending from your server configuration authorization may be required. In this case provide your authentication data, too.
Implementation

Create a VC Model

Start VC Story Board:
http://<hostname>:<port>/VC

Select Model → New Model and type a project name.

Since the iViews to be created in our scenario are intended to be integrated in Guided Procedures (GP) you have to set a compiler flag:

Select Tools → Options
Select tab Compiler and check Add Guided Procedure Information

As you build your model in Storyboard, you can drill down from upper-level model entities down to the lowest-level elements and controls by double click on the corresponding UI container element in the workspace.

Create the iViews and Layers

According to the business process we have to design two pages with different presentation logic. Only IViews can be accessed by GP. So we are starting with two IViews:

1. Event Administration (EventAdmin) – The event administration contains two screens according the user interaction steps, material event selection and material assignment which are modelled by two additional layers:
2. A summary page (ShowSummary) displaying the events and its assigned materials for approval and purchase order handling.
You can add the iViews, as all the remaining UI elements, by dragging and dropping them from the Task panel *Compose Model*.
The picture above illustrates the VC Storyboard with the two IViews in the workspace and all other UI controls of the complete model which are displayed in the tree view in the Task Panel Browse Model on the right.

To add the layers:
1. double click on EventAdmin
2. The iView is now opened in the workspace and the Task Panel will change in this context to provide you with those UI elements which are available for iView design.
3. Select the symbol for the layers and drag and drop it on the workspace. Type name Select Event.
4. Repeat step 3. for layer Assign Material.

**Model the Presentation Logic**

**Data Sources**

From the Task Panel Toolbar select Find Data to open the Find Data Services Task Panel.

Select the system name according to your WS Proxy definition (see above).

Browse the required Web Services, select the desired method, drag and drop it as data service to your workspace.

Note: The data service elements cannot be dropped onto a layer directly. Drop them to the space aside (iView level) and then move them to the layers according to the table below.

<table>
<thead>
<tr>
<th>iView</th>
<th>Layer</th>
<th>Web Service</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Admin</td>
<td>Select Event</td>
<td>EDMHandler</td>
<td>getCoursesByTitlePattern</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>getAllEventsForCourse</td>
</tr>
<tr>
<td></td>
<td>Assign Material</td>
<td>MaterialHandler</td>
<td>getAvailableMaterials</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PurchaseRequisitionHandler</td>
<td>getPurchaseRequisitionForEvent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>createPurchaseRequisitionForEvent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>requestPurchaseRequisitionApproval</td>
</tr>
</tbody>
</table>
Modeling UI Elements and Event Handling

You will now model the UI elements and their relationship by event handling. Most of the work is done, if you simply model the UI elements and the event and data flow as depicted below. If you are a really beginner with VC we recommend to import the model from the demo application at first and compare the steps described below to the original.

You may drag drop the UI elements from task panel at first and connect the flow lines by dragging from an output port from one element to the input port of the next element or you may forward engineer your elements by drag and release from the output port of an element and selecting the next element from the context menu.

Design of IView Assign Materials:
Design of iView ShowSummary:

Configuration

Most of the properties of the UI elements are derived from the metamodels and are automatically set by visual modeling as described above. However, in addition there are certain configuration steps to complete the presentation logic.

In the following will show some exemplary configuration steps in detail to demonstrate the configuration techniques with VC. For completeness refer to the table at the end of the chapter.

Submit search string: To assign the string Value you have entered in the Search form to the submit event, you have to enter a simple formula.

1. Select the submit arrow, right click and select *Configure Element*.

2. In the empty field *Assigned Value* click on the formula symbol and in the pop-up (compare picture right hand) select *Data Fields* --> *Search* --> *title*.
Add fields to a Table View

1. Select the Table View *Events*, right click and select *Configure Element*.

2. Klick in the pop-up window select an input field and type and add the field name.

Add transition between layers

1. Select layer *Select Event*, drag on the green point on top and drop to the layer *Assign Materials*. This create a transition event arrow between the layers.

2. Enter name *assign* for the transition event (do the other way round and create transition event *request_approval*).

Define actions of buttons (example button “Add” of Form View *Assign Buttons*)

1. Select the Form View *Assign Buttons*, right click and select *Configure Element*.

2. Klick , select type Pushbutton and enter name BTN_add.

3. Double click on the field entry and the window Control Properties open up.

4. In the tab *Action* mark *System Action* and select *Insert Row* at system action. This action should be applied to *Assigned Materials*.
Calculate virtual table field in iView ShowSummary Table View Assigned Materials

1. In Table View Assigned Materials add an additional summary field “SUM” of type number
2. In the tab General of the Control Properties Window click on button Formula for the default value.
3. Enter the formula as depicted. To get the field with the correct id select from Data Fields → Assigned Materials

Assign values from available material to assigned materials:

1. In Table View Assigned Materials in task panel Configure double click field currency.
2. In the tab General of the Control Properties Window click on button Formula for the default value.
3. Select Data Fields → Available Materials → currency
4. Repeat the steps before for the field’s description, matId and price as well.

The following table summarizes the configuration steps (values to be changed):

<table>
<thead>
<tr>
<th>Layer/iView</th>
<th>UI element</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Event</td>
<td>Start point</td>
<td>Input Parameter=title, Data type=Text</td>
</tr>
<tr>
<td>Form View Search</td>
<td></td>
<td>Input Field=title, Data type=Text Push Button=SUBMIT properties: choose action, system action=submit form, apply to self</td>
</tr>
<tr>
<td>Data Mapping from Form View Search</td>
<td>assign title form Data View → Search</td>
<td></td>
</tr>
<tr>
<td>Table View Courses</td>
<td>mark all fields visible</td>
<td></td>
</tr>
<tr>
<td>Data Store store</td>
<td>add field STR_Course of type Text add field STR_Course_title of type Text</td>
<td></td>
</tr>
<tr>
<td>Data Mapping from Table View Courses to store</td>
<td>event name=select</td>
<td></td>
</tr>
<tr>
<td><strong>Form View Assign</strong></td>
<td>Push Button=BTN_assign, choose properties action, Custom Action = ASSIGN apply to self</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Data Mappings from Form View Events</strong></td>
<td>Event name=<em>assign (</em> is used as place holder here)</td>
<td></td>
</tr>
</tbody>
</table>
| **Assign Material** | **Form View Event Details**  
make all fields visible except courseId for all visible input fields besides quantity: In property tab general set Disabled=true  
add Expression Box STR_Course and enter formula: STORE@STR_course  
add Expression Box STR_Course_title and enter formula: STORE@STR_course_title |
| **Data Mapping from Form View Event Details** | Event name=*save |
| **Table View Assigned Materials** | Selection mode=multiple  
field quantity: mark visible, default value=1 |
| **Link from Data Element "getpurchaserquisitionforevent" to Table View Assigned Materials** | right click and select Configure Element  
Delete  
Change link to Data Flow |
| **Data Mapping from Table View Assigned Materials** | Event name=*save |
| **Form View Buttons** | Push Button=BTN_assign_materials, choose properties action, Custom Action = REQUEST_APPROVAL, apply to self  
Push Button=BTN_saveall, choose properties action, Custom Action = SAVE, apply to self |
| **Data Mappings from Form View Buttons** | assign eventId form Data Fields → Event Details |
| **Form View Assign Buttons** | Push Button=BTN_add, choose properties action, System Action Insert Row, apply to Table View Assigned Materials, mark insert last  
Push Button=BTN_remove, choose properties action, System Action Delete Row apply to Table View Assigned Materials |
| **ShowSummary** | Start Point |
| **Input Parameter=approvalStatus (Text)** | Input Parameter=eventId (Number, required)  
Input Parameter=poNumber (Text) |
| **Data Mapping From Start Point to Web Services** | Input Field eventId=@eventId (required) |
| Form View Message | Plain Text=msg In control properties edit formula for label: IF(ISNULL(@approvalStatus) OR @approvalStatus=="", "You are requested to approve the displayed materials for the given event", IF(@approvalStatus=="true", "The following purchase order request has been approved", "The following purchase order request has been rejected")) |
| Form View Event | select all fields except courseId  
Editing Mode=Read-only |
| Table View AssignedMaterials | additional calculated field SUM (as already described above by quantity * price)  
Editing Mode=Read-only |
| Form View Purchase Order | Editing Mode=Read-only  
display poNumber only  
and enter visibility condition: NOT ISNULL(@poNumber) |
| End Point | no event |

**Customize the Layout**

After finishing the modeling tasks in the design tab of your workspace you should arrange the UI elements in the layout tab of your workspace.

In this perspective you may check also all the labels and visibilities of the UI controls. Compare to the screen shots in the chapter “Create the iViews and layers” above.

**Deployment**

Select Deploy from Task pane and press deploy.

Depending from the hierarchy level of your actual workspace, you may want to check “Deploy entire model”, if want to deploy the entire model.

The button Deploy activates the compilation and deployment process. Successful deployment is indicated by a Deployed Successfully message. If errors or problems occur, the appropriate messages or warnings are displayed along with the source code. To view the deployed file, click on the Run “<iView Name>“ line.
Testing and Error Analysis

Testing

After deployment you can test your iView simply by clicking Run “<iView name>” in the deployment pane (see screenshot above).

Optionally you can test the functionality of a data service that has been added to an iView separately.

1. In the Design workspace, right-click the data service that represents the function you want to test and choose Test Function.

2. In the Test Function dialogue box you can add the test parameters and execute the data service.

However, it is strongly recommended that you have already tested your services at implementation time and during service enabling (see part I and II of the documentation).

To review the overall UI logic in your iView, Visual Composer provides a tool for tracing dependencies in your diagram. It shows which elements are dependent on other elements:


2. From the Options menu in the Fields task panel, choose Trace Dependencies.

3. In the Design workspace, choose a data service or UI component. Everything on which the selected element depends is marked in pink, while everything depending on the selected component is marked in blue.

Debugging

Once you have deployed the model, you can debug it using the Build Æ Debug the Model option.

In the Debugger task panel you can turn the event monitor on to run a selected event, one step at a time, and/or closely examine data sets within iView components using the Data Inspector. Please refer to Debugging the Model in the SAP Help Portal for more details.
Related Content

- [How to Consume ES-Workplace Enterprise Services in Visual Composer](#) (PDF 245 KB)
- [SAP NetWeaver 2004s Installation Master Guide](#) (PDF 17 MB, requires Service Marketplace login)