Creating Next Generation SAP Analytics Applications with SAP NetWeaver and Macromedia Flex

June 2005

Christophe Coenraets (Macromedia), Vincent Mendicino (SAP),
Natalia Shmoilova (SAP), Dirk Wodtke (SAP)

Abstract. This paper describes how end-user friendly SAP Analytics applications can be created by combining the SAP NetWeaver integration platform and SAP NetWeaver Visual Composer along with the Macromedia Flex (available within SAP NetWeaver). By combining the three technologies users are empowered to visually design application logic and process flows to provide more effective, engaging, end-user experience. The concept is applied to the domain of SAP Analytics but can be used for other application types as well.

.

Copyright Notice

Table of Contents

Introduction

This paper describes the concepts and architecture of SAP Analytics applications in SAP NetWeaver. SAP Analytics applications are a valuable addition to the Business Intelligence offering included in the SAP NetWeaver platform. The implementation of SAP Analytics uses two key technologies:

- SAP NetWeaver Visual Composer
- Macromedia Flex

SAP NetWeaver Visual Composer is a new and innovative SAP NetWeaver component. It provides a model driven approach towards the graphical composition of applications.

Macromedia Flex is a presentation-tier framework and server that enables the development of Rich Internet Applications (RIAs). RIAs combine the responsiveness and richness of desktop software with the broad reach of web applications to deliver a more effective end user experience.

The paper is structured as follows: Section 0 describes the business side of the problem we are solving and explains SAP Analytics. Section 0 gives an overview of the solution. Section 0 describes the technologies which are used in the solution. Section 0 explains the solution in more detail. Section 0 describes an example scenario. Section 0 gives a summary and an outlook.

Statement of the Problem

In a business environment rapid access to relevant business data is essential for efficient decision making. Decision makers demand tools which support them adequately when making their business decisions. They don't have the time to locate the needed information in a Business Information Warehouse application or other Data Services. They expect the information to be displayed directly on their portal pages and to be presented in an intuitive and aesthetically pleasing manner.

SAP Analytics help to solve this problem by collecting relevant information from disparate Data Services, providing an intuitive visualization of the data on consistent screens, and exposing appropriate action options to the user. SAP Analytics applications can further be customized by end users so that they only display the exact subset of the data that is needed and be limited only to collaborative or transactional permissions based on the specific user's role or function. A typical SAP Analytics screen consists of a variety of visualization "widgets" like pie chart, bar chart, ticker, table, pivot table, map, etc. as well as specific application controls interacting with back-end business applications.

SAP Analytics applications are used by corporate decision makers. Usually these are non technical users of the system. Their focus is on interpreting for example a sales pipeline and on knowing when to take action based on the data that is displayed. That's why it is very important to intuitively display the data.

SAP Analytics applications are created by business analysts or consultants. They have the right mixture of knowledge on how to access corporate data in the IT applications and a good understanding of the business domain. Usually, they are not required to have any deeper programming skills.

Analytic applications from SAP have already been around for a number of years. The novelty of the SAP Analytics applications as they are presented here is that they make access to analytic and transactional data easier and are more intuitive to use than traditional analytic applications. While they are functionality wise as powerful as traditional analytic applications the user interface is more efficient, more delightful, and more user centric.

Overview of the Solution

One of the core value propositions of SAP NetWeaver is its simplicity, increased usability, and adaptability. SAP Analytics applications substantiate this value proposition in at least two areas: Allowing for a rapid development of high quality SAP Analytics applications by using Visual Composer and providing an agile and highly usable user interface for SAP Analytics applications by using Macromedia Flex.

Rapid Development
The creation of a SAP Analytics Application by a business analyst or consultant is an easy three step process:

Step 1	The user identifies the Data Service(s) which provide(s) the data which should be displayed on the SAP Analytics Application. Supported Data Services include (but are not limited to) Business Warehouse info cubes, and Data Services which are reachable through JCA compliant resource adapters (BI JDBC Connector, BI ODBO Connector, BI SAP Query Connector, BI XMLA Connector).
Step 2	The user identifies the graphical elements/controls which are used for displaying the data.
Step 3	The user positions the controls on the SAP Analytics Application, connects them with the selected Data Service(s) and uses the Visual Composer deployment facility for generating and deploying the SAP Analytics Application to the Enterprise portal.

Right after its deployment the SAP Analytics Application is immediately available to all Portal users (sufficient Portal role assignments assumed). Users can browse the set of deployed SAP Analytics applications in the Portal Content Directory, select the ones they would like to appear on their portal pages, and customize them to their liking.

It is important to note that when creating a SAP Analytics Application no program code has to be implemented. This ensures a generally good quality of SAP Analytics applications and allows for a smooth re-generation of multiple versions of the same SAP Analytics Application during the creation phase. The creator of the SAP Analytics Application can concentrate on its business aspects and does not have to deal with low level technical difficulties.

Agile and Highly Usable User Interface
Usability studies proved that knowledge workers are better served by user interface technologies which avoid frequent server roundtrips (less wait time for end users means more productive end users) and allow for immediately visualizing effects of data updates users perform (for example, if the user is immediately presented a graphical representation of numeric data she just entered she can simulate the effects of different values for the same data object).

The Macromedia Flex technology minimizes the number of server roundtrips required by delegating processing tasks to the client. Communication between different controls on the same page can be implemented locally on the client. For example, the communication between a table and a pie chart which visualizes the data in the table is also done on the client without accessing the server.

SAP NetWeaver, Visual Composer, Macromedia Flex

1.1 What is SAP NetWeaver?

SAP NetWeaver is an integration platform and the technological foundation for all SAP products since the SAP Business Suite. It is a service oriented application and integration platform, i.e. NetWeaver is the interface between SAP applications and it is their runtime environment. It furthermore interoperates with and can be extended using Microsoft .NET, Sun's J2EE, and Macromedia WebSphere. SAP NetWeaver embraces Internet standards such as HTTP, XML, and Web services. It also enables Enterprise Services Architecture, SAP's blueprint for service-oriented business solutions.

As part of SAP NetWeaver there is an Enterprise Portal which helps create software that brings together all the data and software tools that a person needs to do her job in one consistent user interface.

Furthermore, SAP NetWeaver contains a Business Warehouse (SAP BW). SAP BW is a comprehensive end-to-end data warehouse solution with optimized structures for reporting and analysis. In addition SAP BW includes decision support, online analytical processing, statistical analysis, forecasting, and data mining. SAP BW comes in conjunction with a Business Intelligence (BI) component which provides tools for information integration, so that all transactional and aggregated information is consistent and accurate.

1.2 What is SAP NetWeaver Visual Composer?

SAP NetWeaver Visual Composer is a highly productive, model-driven tool aimed at business analysts and developers for code-free creation of composite applications. It is a powerful design tool that facilitates the creation of content for the Enterprise Portal in SAP NetWeaver, using a visual user interface rather than manually writing code. It provides sophisticated, yet simple-to-use tools for creating portal pages and iViews that process data from back-end systems.

Visual Composer operates on top of the Enterprise Portal, utilizing the portal's connector-framework interfaces to allow users to access a range of Data Services, including SAP and third-party enterprise systems. This is done through special connectors which give access to the Business Warehouse and any Open/JDBC Data Service. Because Visual Composer is a fully Web-based application, business consultants can interact with business users (content experts) to build or customize portal pages and iViews as needed, accessing the software from any machine.

Visual Composer Architecture

The general architecture of Visual Composer is depicted in the following diagram. The Visual Composer compiler lets users generate portal pages and iViews that can be deployed to the Enterprise Portal using a standard deploy procedure. Once deployed, the pages and iViews created by Visual Composer can retrieve data from the SAP Enterprise Portal runtime connectivity framework and display this information just like any other hand-coded portal pages.

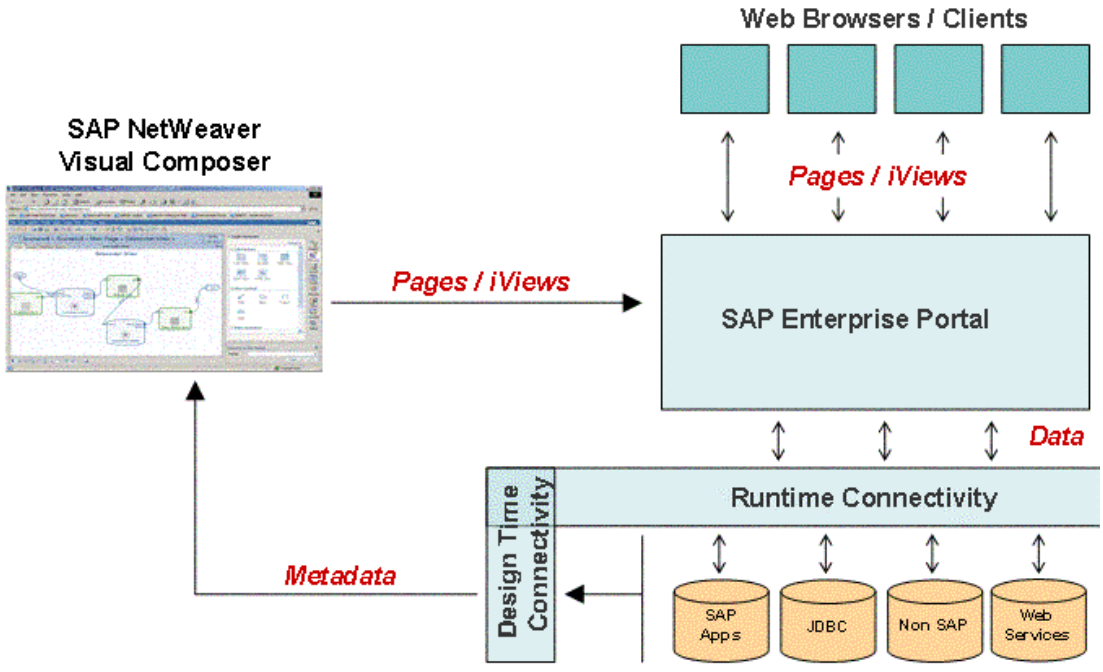

Figure: SAP NetWeaver Visual Composer Architecture

Visual Composer Modeling Concepts

Modeling concepts in Visual Composer are very intuitive: Business applications can be created using an easy drag-drop-connect paradigm. In addition, the application diagram can be viewed in various perspectives:

- Model view – a graphical representation of dataflow, visualization and events.
- WYSIWYG view – allows users to perfect and fine-tune user interface.
- Preview view – allows for test-driving the result before compiling it and deploying to SAP NetWeaver Portal.

The following two screenshots show the Model View and the Preview representation of a simple Visual Composer model.

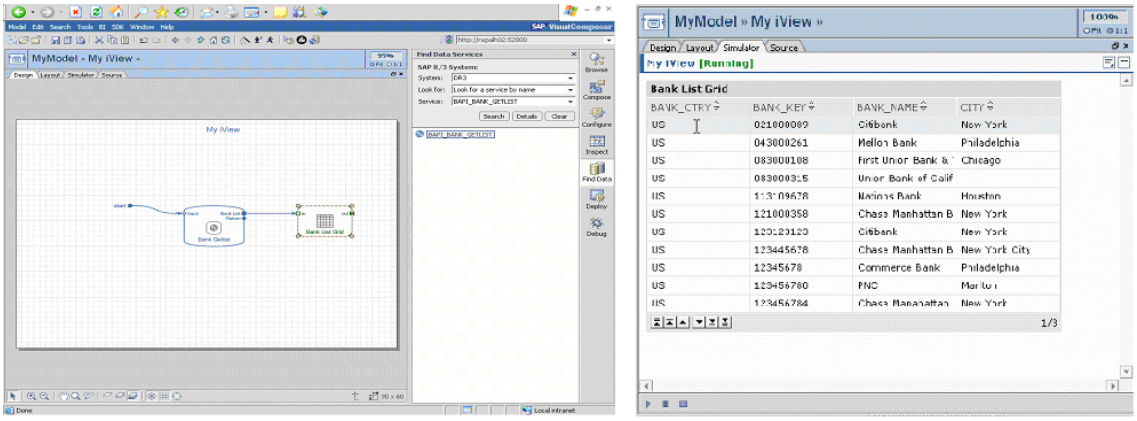

Figure: SAP NetWeaver Visual Composer Modeling View And Preview

Visual Composer Backend Connectivity

Visual Composer connects with a variety of Data Services. The most important Data Service for SAP Analytics is the SAP Business Warehouse (SAP BW).

The Business Intelligence (BI) Kit which is delivered together with Visual Composer is a tool which gives application developers access to the analytical content both of the SAP BW and of external Data Services through BI JDBC, BI SAP Query, BI ODBO, BI XMLA, BW Web services connectors, and native BW Web Applications.

By using the BI Content Wizard which is also included in Visual Composer users can build new queries in a guided way based on an extensive catalog of templates or on a free form query designer.

Visual Composer UI Generation

At its core Visual Composer stores models in an XML representation, the Visual Composer Language. The following diagram gives an example of the VCL representation of a Visual Composer model.

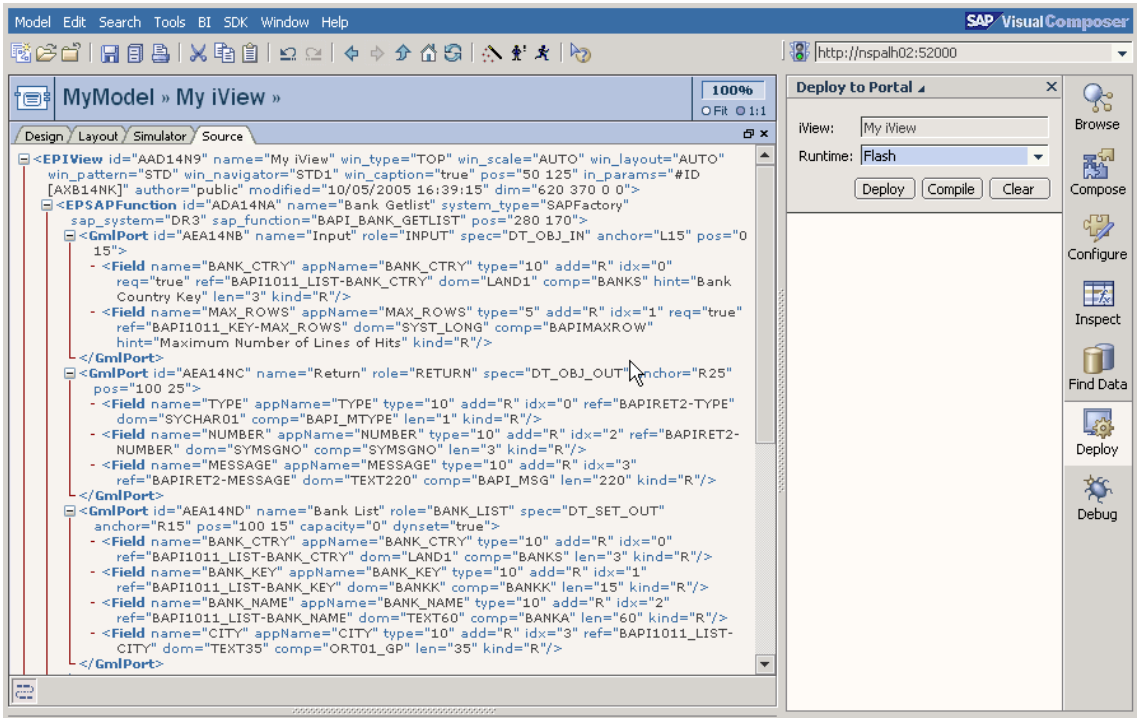

Figure: SAP NetWeaver Visual Composer Language

Following the principle "Model once – run everywhere", Visual Composer models are abstract and independent from concrete UI technologies. Only at the later stage, when the modeling part is complete, the Visual Composer user decides on the UI technology for generating the actual result. This can be Flash (which represents Flex), SAP's adoption of HTML to business applications, DHTML or SAP's WebDynpro UI technology.

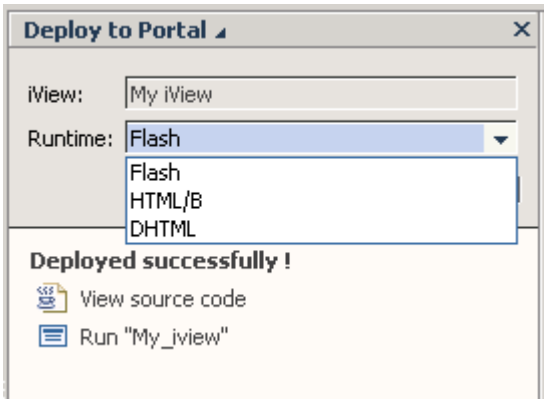

Figure: SAP NetWeaver Visual Composer Supported UI Technologies

During the deployment, the Visual Composer is compiled into executable code, corresponding to the selected UI technology.

1.3 What is Macromedia Flex?

Macromedia Flex is a presentation-tier framework and server that enables the development and deployment of Rich Internet Applications. These applications combine

the richness and responsiveness of desktop applications with the broad reach of web applications.

Flex overcomes the traditional limitations of HTML-based user interfaces. Flex applications support rich user interface components and direct object manipulation such as drag-and-drop. They provide visual cues and transitions to help the user make sense of state and data changes in the application. Flex applications can also support real time data push and rich media streaming.

Flex is used to improve the user experience in a wide range of applications, including dashboard, business process automation, self-service, and commerce applications.

Flex applications are stateful. In other words, the state of the application can be maintained at the client-side. Flex also enables client-side data manipulation (for example, sorting and filtering) and caching. These features can significantly minimize the number of server roundtrips. This results in more responsive applications that also reduce the load on the network and on the server; see the following figure for an illustration of this.

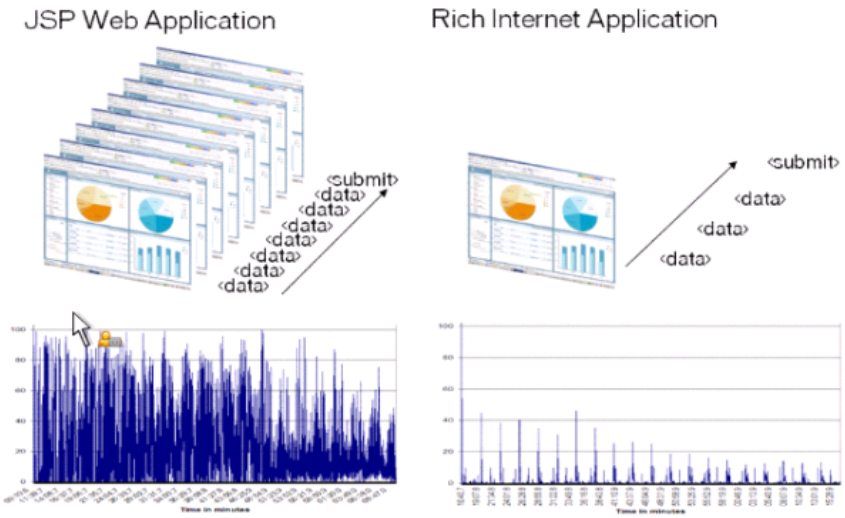

Figure: Network usage comparison between traditional JSP Web Applications and Rich Internet Applications

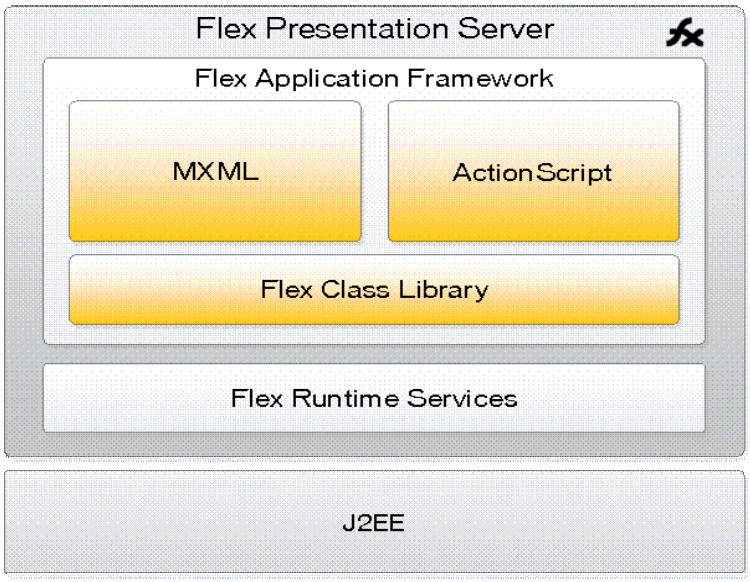

Figure: Flex Presentation Server Architecture

Flex consists of a standards-based, highly productive programming model, an extensive class library of user interface components, and runtime services for backend integration.

The Flex runtime services run on top of Java application servers, including the SAP NetWeaver J2EE engine.

Standards-based, highly productive programming model
Rich Internet Applications built on the Flash platform, and their associated benefits, are not new. However, until recently, these applications could only be built using the traditional Flash IDE whose primary focus is not on traditional application development. Flex addresses the specific requirements of enterprise developers by providing a programming model that supports standard software engineering methodologies and design patterns, and that integrates with the existing enterprise infrastructure. Flex enables the development of large scale applications that are easy to develop, debug, and maintain.

The Flex programming model is based on the combination of MXML and ActionScript 2.0. MXML is a declarative, XML language used to define the user interface of the application. ActionScript 2.0 is an ECMAScript-compliant strongly-typed and object-oriented language that is used to implement the non-visual aspects of the application (client-side logic).

The source code of the Flex application (XML documents and ActionScript classes) is compiled into Flash bytecode (SWF file). This allows the Flex application to be delivered to the ubiquitous Flash Player running inside the browser at the client-side using the traditional lightweight, cross-operating system, and cross-browser Web deployment model.

Class Library
Flex features an extensive library of user interface components, including DataGrid, Tree, TabNavigator, Accordion, Menu, media controllers, and a wide variety of charting

components. Flex components are customizable – using cascading style sheets (CSS). New components can also be created from scratch, by extending existing components (using traditional object-oriented inheritance), or by aggregating other components. Flex components are available as tags in the MXML language, and can also be instantiated programmatically in ActionScript.

Runtime Services

Flex focuses on the presentation tier of the application, and provides data services to connect to the server using SOAP-based web services, XML over HTTP, and remote method invocation into Java objects. Visual Composer leverages these services to provide transparent connectivity to SAP and non-SAP, OLAP and Relational Data Services. Other runtime services available in Flex include dynamic compilation, caching, as well as integration with the session management and security infrastructure of the underlying application server.

Example

The following example shows a simple Flex application and the corresponding MXML source file:

Figure: Flex example application

```
<?xml version="1.0" encoding="utf-8"?>

<mx:Application xmlns:mx="http://www.macromedia.com/2003/mxml">

<mx:WebService id="ws" wsdl="inventory.wsdl">
        <mx:operation name="getList"/>
</mx:WebService>

<mx:Panel title="Product List" width="100%" height="100%">

        <mx:DataGrid id="dg" dataProvider="{ws.getList.result}">
                <mx:columns>
                <mx:Array>
                <mx:DataGridColumn columnName="name" headerText="Name"/>
                <mx:DataGridColumn columnName="price" headerText="Price"/>
                </mx:Array>
                </mx:columns>
        </mx:DataGrid>

        <mx:ControlBar>
                <mx:Button label="Get Data" click="ws.getList()"/>
        </mx:ControlBar>

</mx:Panel>

</mx:Application>
```

Code highlights:

The <mx:WebService> tag is used to define this application as a client for a specific web service.

The <mx:DataGrid> is an example of a powerful component available in the Flex class library. It is sortable, supports drag-and-drop, and can be made editable if required. Databinding is used in the dataProvider attribute to populate the grid based on the result of the invocation of the getList method in the web service.

In the “click” event handler, of the button, the getList() method of the web service is invoked using ActionScript’s standard object-oriented dot notation.

Solution in Detail

1.4 Concept

SAP Analytics development in Visual Composer is targeted towards business analysts and consultants who do not require (and may not have) programming skills. Their focus is on knowledge of business processes, systems, and data. This is accommodated by a model driven approach where the actual programming code is generated from an abstract model.

By building a model, the SAP Analytics developer can produce a fully functional analytics application in a user friendly drag and drop environment with no coding effort. By not actually coding, the risk of mistakes is limited and in a short time an appealing analytic application can be created.

Once a model is built, editing the model can be done very easily. Replacing a Data Service or changing the visualization from a table to a chart can be done with a simple drag and drop. This provides flexibility, allowing the business analyst to quickly prototype the applications to meet a client’s needs.

Once Data Services and visualization have been decided it is possible to generate the SAP Analytics Application as a Flex application and deploy the application to the Enterprise Portal.

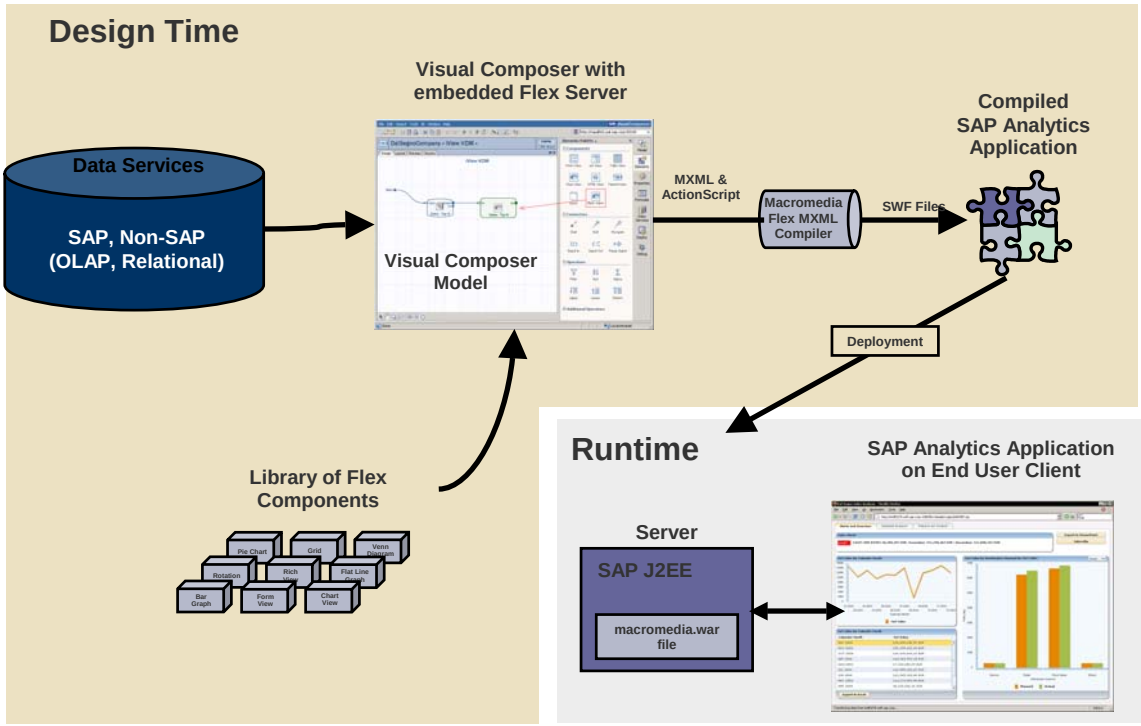

Figure: From Creating a Visual Composer Model to Running a SAP Analytics Application

The previous diagram gives a summary of the components being used for creating and running a SAP Analytics Application. In the Design Time after a Data Service and a Flex component have been selected they are linked with each other. The set of selected Data Services and controls form the Visual Composer model. The model is represented in VCL and during the generation step the model is compiled into MXML and ActionScript files. As an optimization, in a second compilation step it can be compiled into a SWF file and copied onto the NetWeaver application server which will host the analytics application. At Runtime, upon navigating to the corresponding Enterprise Portal page the SAP Analytics Application is displayed in the browser of the end user using the Flash Player browser plug-in.

The steps for creating a SAP Analytics Application within SAP NetWeaver Visual Composer are illustrated in the following figure.

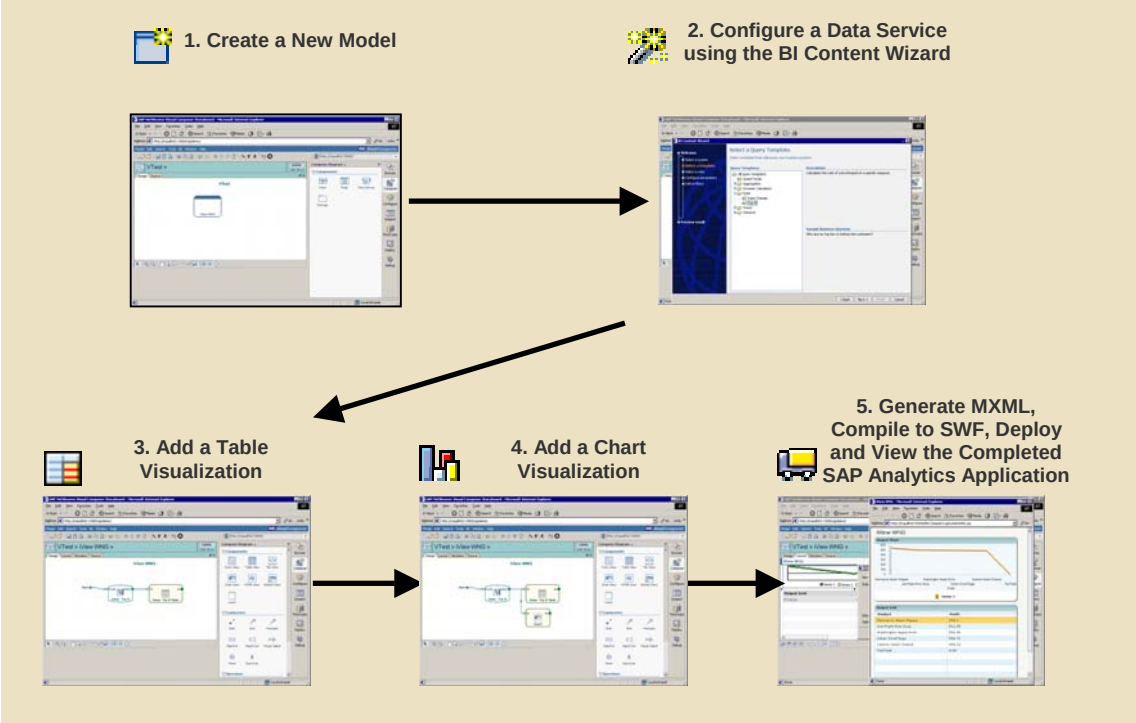

Figure: From Creating a Visual Composer Model to Running a SAP Analytics Application

The steps are as follows:

1. A new model for the SAP Analytics Application is created.
2. A Data Service is selected. For example the BI Content Wizard is used which streamlines the Data Service connection configuration process for BI specific Data Services. During the process of creating the BI Data Service the user will be asked to select from a list of queries and data cubes. Once the process is complete the BI Data Service behaves as any Data Service in Visual Composer.
3. The table control comes as the default visualization for the BI Data Service; further configuration can be done to the table.
4. For quickly analyzing a result set it is often easier to additionally opt for a chart based visualization. This is done by just dragging the output of the BI Data Service and select “Add Chart”. Any configuration can then be made by modifying the chart’s properties.
5. Finally, the newly created SAP Analytics Application’s MXML is generated, compiled into a SWF file and deployed onto the Enterprise Portal. Immediately it is available via the web browser.

1.5 Architecture

The Visual Composer has both an IDE component and a server component. The IDE component consists of a modeler, Data Service configuration tool, layout designer, visualization components and source generators. The server component runs on top of the J2EE Engine of the SAP NetWeaver Web Application server with SAP and Macromedia Flex components as well as the actual SAP Analytics running within it.

The IDE's visualization components include tables, charts and form inputs which are all configurable using simple dropdowns, input fields and checkboxes. Data Services can be configured by using a wizard or by selecting tables from a list of available Data Services provided by the Enterprise Portal in the case of a relational database Data Service. The modeler handles the connections between the Data Services and visualizations. Data Service data can easily be directed to a variety of visualizations, events and event actions can also be configured with the modeler.

Once visualizations and Data Services are modeled the layout component is used to configure the positioning of the visualization elements. The layout component features simulated visualization components which can be resized and positioned on a simulated canvas to ensure proper layout before deployment to the Visual Composer server. When the layout has been decided on the deployment is started. The deployment process includes the generation of Flex code and compilation into a SWF file.

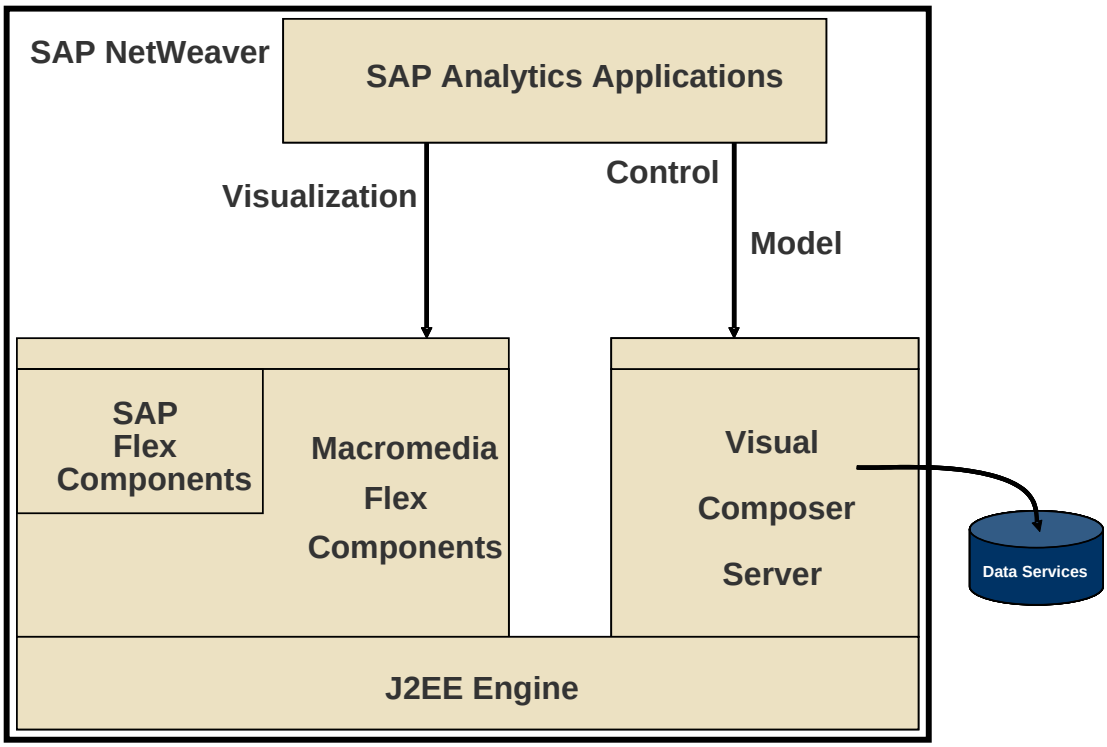

Figure: Interaction of SAP NetWeaver and Macromedia components

The server component of Visual Composer is built on top of the SAP NetWeaver platform and runs on top of the J2EE Engine. All Data Service requests are routed through the Visual Composer server. Above the SAP J2EE Engine are the standard Macromedia Flex components. These are the Macromedia tables, charts, form elements and compiler that come standard with a Flex installation. SAP has extended Macromedia components for additional functionality and customization, this extension sits alongside the Macromedia components.

The visualization components form the base of a SAP Analytics Application. On a web browser request for a SAP Analytics Application, both the component layer and SAP Analytics Application get compiled by the Macromedia compiler into a SWF file. The

SWF file is then shipped to the client where the application is executed by the Flash Player.

Example

An authentic – admittedly simplified – scenario where a SAP Analytics Application greatly helps to solve a business problem is the processing of sales orders in an enterprise. At some point during the processing of a sales order the credit manager of the seller company is involved. Her job is to decide whether a sales order which is blocked due to a failed credit limit check can be released or should be canceled. To come to a decision the credit manager needs to have an overview of the most important customer data and key performance indicators. The relevant data is potentially spread across multiple systems. Without the SAP Analytics Application it is hard for the credit manager to collect all the data needed.

By using the SAP Analytics Application the credit manager is able to identify those customers who are facing financial hardships. Those customers represent a major risk for the seller and their purchase orders should be cancelled.

1.6 Creating the SAP Analytics Application "Sales Order Credit Check"

The first step when creating the SAP Analytics Application is to identify all Data Services which provide the data that will be displayed on the application screen. In the case of the Sales Order Credit Check the creator selects the following Data Services:

- "Blocked Order List": This is a BAPI which returns the set of blocked orders, `ZRFC_BLK_SALESORDER_GETLIST`.
- "Exposure List": This is a Business Warehouse Query View which returns information on which products the customer has ordered so far grouped by Business Unit, `DD_FIN_EWL_BLANK`
- "Customer Detail": This is a Query View which returns aggregated customer detail information from the Business Warehouse, `OCDM_C0_Q0002`
- "Customer Score History": This is a Query View which returns the customer score history from the Business Warehouse, `ZOCDM_DS02_Q0002_NUM`

The second step is to identify the controls which are used for displaying the data and associate them with Data Services. For the first Data Service the creator of the application chooses a selectable table control. The output port of the Data Service is connected with the input port of the table control. This is for filling the table control with the data from the Data Service. Furthermore, the input port of the control which contains the buttons "Release Order" and "Cancel Order" is connected with the output port of the table control. By doing this for example a mouse click on "Release Order" releases the order which is selected in the table control.

For displaying the customer's exposure list the input port of the second Data Service is also connected with the output port of the table control. For displaying the information the input ports of a pie chart control and a table control are connected with the output port

of the Data Service. For displaying further exposure details the input port of a detail control is connected with the output port of the table control.

For displaying customer detail data the third Data Service needs both business partner information. By connecting the input port of the Data Service with the output port of the second Data Service the business partner information is propagated. What is left is to connect the input port of the detail control with the output port of the Data Service.

The last mapping needs to be defined for displaying the customer score information. For this the input port of the fourth Data Service is connected with the output port of the third Data Service and the input port of a pie chart control is connected with the output port of the Data Service.

The following figure illustrates the mappings as they have been created in Visual Composer.

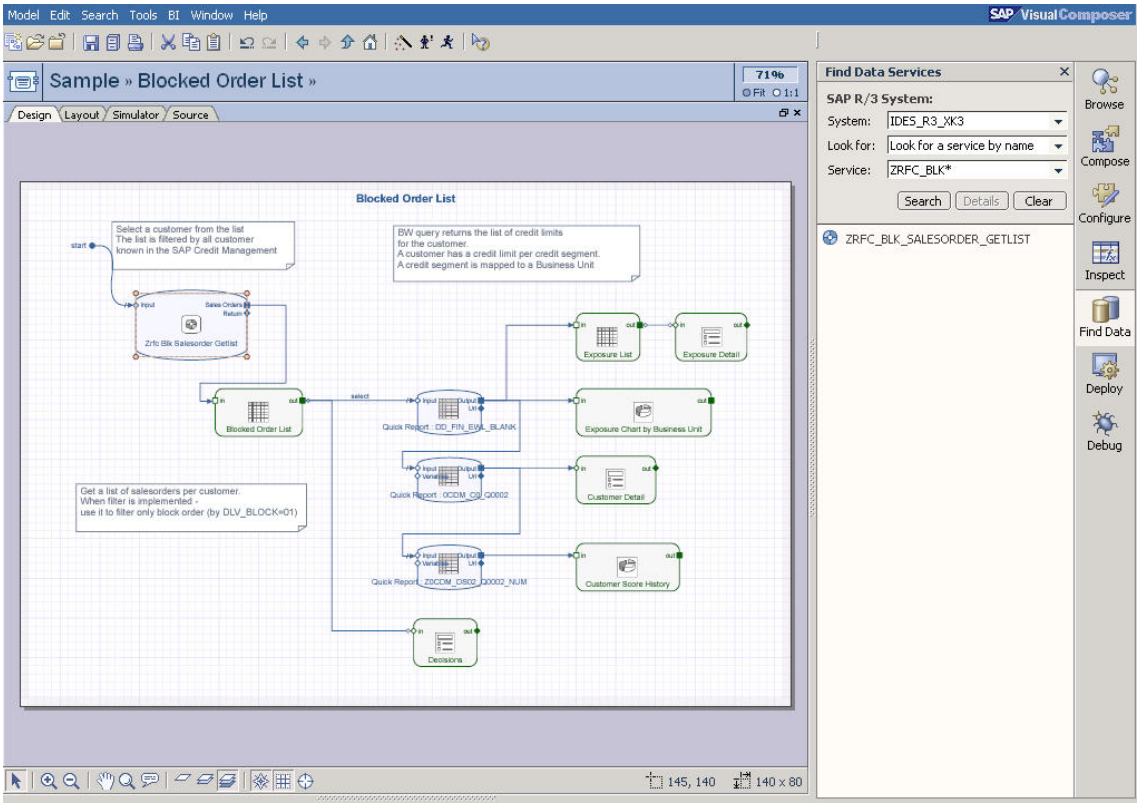

Figure: Creation of the SAP Analytics Application "Sales Order Credit Check" in SAP NetWeaver Visual Composer

On the right hand side the screenshot shows the Data Service browser with the BAPI for the first Data Service being selected. All operations can be performed by using drag & drop. No code had to be implemented by the creator of the application.

The last step in the process of creating a SAP Analytics Application is to graphically arrange the controls into an intuitive order.

1.7 Running the SAP Analytics Application "Sales Order Credit Check"

Figure: Running the SAP Analytics Application "Sales Order Credit Check"

The figure shows the result of the SAP Analytics Application creation. At the top of the application screen there is the list of blocked orders. The buttons below the table are linked with the selected order from the list. Upon selecting an order details of the customer who initiated the order are displayed in "Customer Detail" underneath the buttons. Also the Exposure List and the Customer Score History are updated with customer information and displayed. Depending on which Business Unit is selected in the Exposure List the bar chart Exposure Chart by Business Unit is immediately updated. At the same time the section Exposure Details including the gauge is automatically updated.

In this SAP Analytics Application the usability and performance benefits of using Flex really are apparent as opposed to a DHTML implementation. This first table with the list of Blocked Sales Orders can be sorted by clicking on any column header, without delay the entire table is sorted on the client. If this application were implementation in

DHTML in all likelihood this sort operation would require a server round trip and cause the entire page to flicker. Besides client sorting, another important benefit is the selection of a Blocked Order entry from the table and the subsequent display of details for that Blocked Order. In this case there is a round trip, as would happen in a DHTML implementation, but there will be no page refresh. All charts, dials, gauges and tables are transitioned into their next state without the flicker of a refresh.

Summary

This paper described how end user friendly SAP Analytics applications can be created by combining the SAP NetWeaver integration platform and SAP NetWeaver Visual Composer with Macromedia's Flex presentation system. Using Macromedia's Flex presentation system was essential in SAP's effort to quickly develop the SAP Visual Composer's code generator for SAP Analytics as opposed to using native Flash. By combining the three technologies users are empowered to visually design application logic and process flows to provide more effective, engaging, end-user experience. With this concept and these technologies it was possible to create more than 200 SAP Analytics applications with a few weeks of work. Feedback received from consultants who created the applications and customers who test drove some of the applications that have already been built was overwhelmingly positive.

SAP Analytics, Powered by SAP NetWeaver, will be shipped with an upcoming SAP NetWeaver version. The next version will also come with an updated Flex 1.5 version which provides further optimizations to make SAP Analytics even more effective. Also the SAP NetWeaver J2EE engine has been certified by Macromedia as a supported platform to run Flex applications. This means that SAP developers can also leverage Flex in their traditional development environment, outside of Visual Composer, to provide rich user interfaces to SAP applications.