Adobe Flex Overview

David Roossien
CS658
1/30/2009

Adobe Flex provides a framework of classes for developing applications that can dynamically exchange data with a web server and present data to the user without reloading a web page. Flex applications are the presentation or view tier in a MVC architecture. An interesting example of a Flex application is Yahoo maps.

Flex provides a user interface markup language called MXML for designing the layout of the interface. MXML provides a long list of useful classes for presenting data and generating events to request data. A notable class is the DataGrid, which provides a simple way to display data in a spreadsheet-like grid. Users can edit and rearrange the grid by dragging and dropping the columns in the order they choose.

MXML is a proprietary language that was originally developed by Macromedia, which was then purchased by Adobe. It is a declarative language, similar to HTML, but it provides more structure because it is well formed XML. There are many training tools available with excellent example and a one week training video series titled "Flex In A Week" (http://www.adobe.com/devnet/flex/videotraining/).

ActionScript provides a Javascript-like programming language for capturing events from the MXML components, making requests to the Web Service for data, and loading response data into the MXML components.

Flex applications runs in either Adobe Flash player or as desktop applications using Adobe Integrated Runtime (AIR). AIR applications are compiled into an executable binary just like any other desktop application.

Integrated Development Environment

Adobe provides a free SDK that allows you to build Flex applications from the command line. Adobe also created a development environment called Flex Builder that is based on Eclipse. There is also an Eclipse plug-in version. The IDE provides a Design and Source view for development as well as debugging tools. When you compile a Flex application in the Flex Builder it creates a SWF "swift file" that can be executed by Flash or AIR. Compiling also creates an HTML file that includes a link to the SWF.

Not a server technology

Adobe Flex does not provide Web Services, rather it requests them and consumes them in many different ways. Flex does not provide support for connecting to a database, rather, it relies on a Web Server to provide its data. One of the goals of Flex is to make it

possible to interact with most of the major Web Service technologies that are popular today. Flex provides a way to interface with servers running with the follow languages:

- JSP
- ASP
- PHP
- ColdFusion

Data Access Components

Both Flash and Flex applications dynamically request and receive updates from server using a **Service Oriented Architecture using Remote Procedure Calls**. Data access components can read and write data to the server and be written in either ActionScript or MXML. Three types of Data Access components:

- HTTP via HTTPService component (send() and get response)
- SOAP via WebService component
- AMF (Action Message Format) Remoting via RemoteObject component

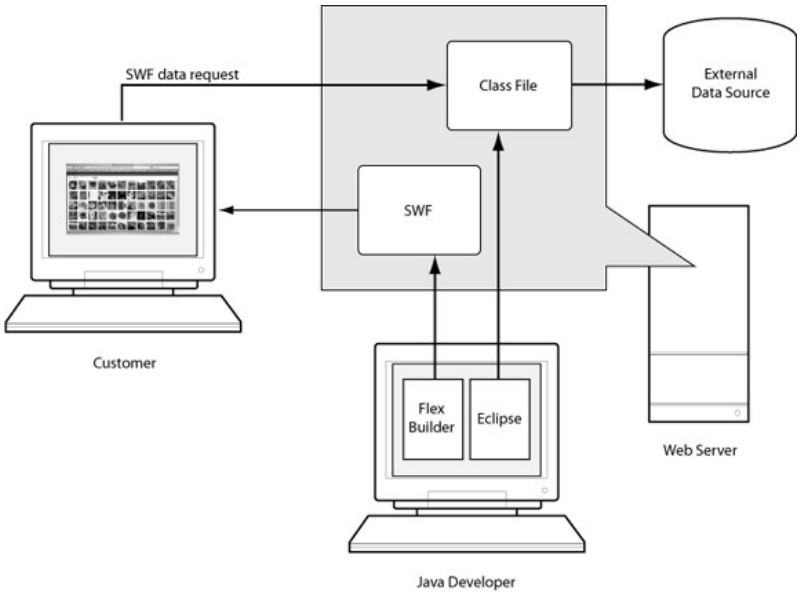
Of these three data access components, AMF is most interesting because it provides a higher performance option due to its compact binary protocol.

BladeDS

Adobe also developed a product called BladeDS Server to make it easier for Java web server applications to communicate with Flex applications. BladeDS uses Java Remoting (RMI) and requires that you run a BladeDS server to capture incoming requests from Flex applications, marshal those request to the Java Web Service, and respond to the Flex application. BladeDS is based on the AMF compact binary protocol, therefore it is high performance. It's

Architecture

As you might expect, the architecture for Flex is pretty straightforward and is similar to other traditional web server applications. A web client navigates to a web page, which references a SWF file that is available somewhere on a web server. The client loads the SWF into its Flash engine. From that point on the Flash application requests data from the configured web service running on a web server. The web server gets its data from its database, files and responds to the Flash application. The below figure shows some of this interaction for a typical Java web server and Flash application.



Developing Applications

Developing small applications for Flash is simple using the "Flex In A Week" tutorials. Developing AIR applications is more complicated because the tutorial requires you to download and install Adobe LiveCycle Web Server product. This product only functions on a server operating system like Windows Server. This makes it impossible for AIR test applications to be developed on non-server machines. There is a free trial for the LiveCycle product, as well as a free "production" version that can be used on single CPU installations. The free production version can be used by developers to develop AIR applications.

Conclusions

Flex applications are well suited to requesting data from a server, handling the response and presenting the data to users. MXML and ActionScript are simple to understand, but it will take a developer some time to become acquainted with the vast number of MXML objects. The development environment is well organized and documented with extensive training available. Developing Flash applications is simple for anyone with a non-server operating system. Developing AIR applications requires setting up a web-server and the tutorials don't do a good job of explaining how to use Adobe's LiveCycle ES product. The LiveCycle ES product requires a server operating system, so developers will need to spend extra time setting that up.

Selecting the most suitable Data Access Component is probably the most important decision a developer can make. Of the 3 major options the Action Message Format is the highest performance option because it uses a compact binary format. A BladeDS Server is an interesting option that allows use of the Action Message Format for high

performance and potentially allows existing web server objects to be bridged to Flex applications. A developer would probably need to prototype an application before deciding which messaging technique to use.

References:

http://www.adobe.com/devnet/flex
http://www.adobe.com/devnet/livecycle/
http://www.adobe.com/devnet/livecycle/trial/
http://www.adobe.com/devnet/flex/articles/paradigm.html
http://opensource.adobe.com/wiki/display/blazeds/Overview