Chapter 9
Case Studies Using JavaScript-Based Frameworks

ABSTRACT

In development of thin-client applications, it is a common practice to use server-side technologies in order to create data and business logic back-ends and client side-technologies to create lightweight HyperText Markup Language (HTML)-based front-ends. In the development of Web 2.0 applications, the data and business logic back-ends are typically built on top of third-party Web services. In this context, Simple Object Access Protocol (SOAP) has been traditionally used as the standard communication protocol for eXtensible Markup Language (XML)-based Web services. This chapter presents a review of the support for invoking SOAP-based Web services using Java; then, it discusses the development, using jQuery, Prototype, Dojo, and Java Server Pages (JSP), of different thin-client applications based on third-party SOAP Web services by means of a series of case studies to exemplify the use of some User Interface (UI) patterns for accomplishing rich design principles such as stay on the page and use transitions.

1. INTRODUCTION

Case studies are very important to present real and practical applications about a particular topic. RIAs (Rich Internet Applications) are taking a more important role on Web development, since enterprises are concerned about improving their Web applications to be accessible and have a good presentation. There are several technologies for developing RIAs, such as Adobe® Flex® or Microsoft® Silverlight® to mention but a few, but they represent a high cost of development by having to pay expensive development licenses or purchasing their expensive development suites. Usually, users cannot afford these high costs; because the best option is to use open source alternatives, and, in the RIAs development, JavaScript-based frameworks are the best solution.

It is very important for a developer to increase the development options; and JavaScript could be a formidable option, since it reduces both costs and learning curve. There are different JavaScript-based frameworks such as Dojo, jQuery, Mootools, Qooxdoo, Prototype, Rico, Sencha® ExtJS®, X-Library, GWT (Google Web Toolkit™), Cappuccino, SproutCore, Spry, midori and YUI Library, among others. This chapter presents four different applications developed by using JavaScript-based
technology. The JavaScript-based frameworks used for the development of these applications are jQuery, Prototype and Dojo. These frameworks were selected according to 1) their usefulness for developing JavaScript-based applications in particular and 2) the ability to use two frameworks at once without causing development problems between them. The case studies using JavaScript-based frameworks include different important factors about RIAs development, such as multimedia support, AOP (Aspect Oriented Programming) support, design patterns support, and UI (User Interface) patterns support.

Applications developed in this chapter were chosen because they enable to easily illustrate the use and features of the JavaScript-based frameworks, as well as the characteristics of RIAs, such as avoiding page changes and keeping the focus on the main screen, among some others.

The RIAs developed were:

1. A world atlas that is a system for displaying information about all countries in the world. This application uses multimedia support for displaying related videos and improving the user experience. It also permits loading information on the same Web page. This application exemplifies the use of UI patterns, specifically interaction patterns, through the use of elements such as progress indicator.

2. A document indexing system enabling to search and display results in just one interface. This application provides information about indexed files showing elements such as a tag cloud and a reputation system in order to exemplify the UI patterns implementation.

3. An e-procurement system of medical supplies implemented in a hospital. This application searches for medical supplies of different US (United States) providers. For doing this task, the main interface was designed to show results within a LightBox element in order to avoid page changes and keep the user’s attention on the same interface. Also, auto complete and accordion elements were included to show the UI patterns support.

4. A meta-search engine that represents a search engine showing results in the same input interface. The carousel element was used to exemplify the implementation of UI patterns and an example of the use and implementation of AOP support is presented.

The chapter presents a brief state-of-the-art analysis of each sample application. The analysis is aimed at emphasizing the distinguishing features of the applications developed.

2. DEVELOPING SOAP WEB SERVICES-BASED APPLICATIONS

SOAP (Simple Object Access Protocol) is a simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralized and distributed environment using XML (eXtensible Markup Language). It consists of three parts:

1. The SOAP envelope construct defines an overall framework for expressing what is in a message, who should deal with the message, and whether the message is optional or mandatory.

2. The SOAP encoding rules define a serialization mechanism that can be used to exchange instances of application-defined data types.

3. The SOAP RPC (Remote Procedure Call) representation defines a convention that can be used to represent remote procedure calls and responses.

SOAP messages are fundamentally one-way transmissions from a sender to a receiver; however, they are often combined in order to implement patterns such as request/response. According to the above, SOAP messages can take advantage of the well-known connection model and message