The Use of Adobe Flex in Combination with Java EE Technology on the Example of Ticket Booking System

Przemysław Juszkiewicz, Bartosz Sakowicz, Piotr Mazur, Andrzej Napieralski

Abstract – **The article presents the possibility of building Rich Internet Applications using Flex technology as well as a method of connecting them with Java EE applications based on a Spring framework. As an example, a ticket booking system was created. The most important issues related to rich internet applications and possibilities of used technologies were shown basing on this system. The application was elaborated owning to usage of the latest open-source technologies.**

Keywords – **Rich Internet Application, Flex, BlazeDS, PureMVC**

I. Introduction

The dynamic growth of the Internet over the past several years has contributed to build of a new type of applications - page-based applications [4]. In applications of this type all the data and operations are carried out in one place, which significantly decreases the cost of updating and modernization [8].

However, this solution proved to be not quite perfect, and the reason for this restriction was simple and limited user interface based on HTML technology. Despite the development of HTML language and the use of Dynamic HTML elements (DHTML), the solution was still not sufficient. The incompatibility of this type of application in different browsers forced the developers to create multiple versions of applications for different browsers running on different operating systems.

The solution to the problems of building a business applications has become **Rich Internet Applications** (RIA applications) [9].

One of the main aim of RIAs are moving away from page-based applications, reducing amount of data needed to be transferred, provide a simple application status service, providing an interface known from the normal desktop applications and the ability to operate without connecting to the network:

– departure from page-based applications causes that the page is not generated each time the user performs an operation , and it has directly affects on the amount of data transferred from server to client, pmaz, napier} @dmcs.pl.

– Rich Internet Applications use the resources of user computer, this situation makes that the application state may be stored in RAM, unlike the use of stateless HTTP protocol.

Rich Internet Applications are attractive to users and to Internet Service Providers (ISPs). These applications reduce server load, network traffic and data load time. It does not restrict developers when they create the application interface and provide the same feel and look in different environments [7].

II. Technologies and Tools Used to Develop the System

To build the system authors used the open-source technologies and tools. The basic technology for building the client application is Adobe Flex and PureMVC (most known MVC Flex application framework). Server side application was built using Java and the Spring application framework. This makes the system more flexible and easy to expand. Communication between client and server application is based on Spring BlazeDS Integration project (SBI) and BlazeDS server.

III. Description of the Created System

Ticket booking system is based on two technologies, Java EE and Adobe Flex. Both technologies are constantly and dynamically developed, and Java EE is currently one of the most commonly used technology for building business applications [3].

The system was built as desktop application. The user who wants to install and run it needs the **Adobe AIR** platform. It is also possible to build system as a web application which uses web browser and **Adobe Flash Player** plugin to run.

Manuscript received November 09, 2011.
Katedra Mikroelektroniki I Technik Informatycznych
ul. Wolczanska 221/223 budynek B18, 90-924 Lodz, POLSKA
al. Politechniki 11, 90-924 Lodz, POLSKA
NIP 727-002-18-95
tel. +48 (42) 631 26 45 faks +48 (42) 636 03 27

The main aim of system is to allow reservation of elements (called main elements in the system). The system is designed for booking abstract elements that can be represented, for example, by the film, concert, artistic events, etc.. Each element consists of a description, name, registration type which is necessary for booking element (the system operator depending on the type of registration has to create new client account, or just enter the required informations). In addition, each element has a list of available sites for booking and date with the hours in which it is available. These data are needed to complete the booking process, a combination of dates, places, and an element defines a single reservation.

The system enables full management of the elements, users of the system (divided on the operators and administrators), customers and bookings. Additionally, the system allows to view system statistics, conduct the correspondence between the users of the system and print single bookings.

The system consists of two cooperating parts:

– server side is part of an Java EE application running on application server and it is build using **Spring** framework [10-12],

– client side is built in **Adobe Flex** technology in conjunction with the lightweight **PureMVC** framework that provides a simple application structure based on three elements: model, view and controller.

The cooperation of both server and client side is possible thanks to BlazeDS Spring Integration project. The project uses the BlazeDS server and gives it characteristics of Spring framework. This allows an application built in Flex to use the benefits provided by Spring framework and advantages of its use.

The client application uses two types of services provided by BlazeDS to communicate with the server: RemotingService (RPC - Remote Procedure Call) and MessagingService. RemotingService provides remote procedure call, and MessagingService provides the possibility of sending messages across multiple clients connected to the server.

– Remote Procedure Call is a service that is used to carry out all operations on data including add, delete, edit, and booking of the main elements, management of reservations, users and customers of the system, login to the system and loading a data for statistics.

– MessagingService is a service that is used to inform client applications about changes in the system. Changes cause a series of events and the initiation of operations aimed at synchronizing the state of the client application with the data on server. An example of such behaviour is to automatically log off the user whose account has been blocked by system administrator. At the time of the lock server sends to all client applications message about user who was blocked. The application, in which the blocked user is currently logged on, automatically logout and return to the login screen.

PostgreSQL database was used by authors for storing data and give them the relation character. Thanks to usage of Spring framework and Hibernate persistence API data has the relational structure and it can be mapped to the object model and then send to the client application in that form.

Application architecture is shown in Fig. 1.

```
@Entity
@Table(name="users_details", schema="public")
public class UsersDetails implements Serializable {
        @Column(name="id", nullable=false)
        @Id
        private int id;

        @Column(name="name", nullable=true, length=255)
        private String name;
```

Fig. 1. Parts of POJO class with mappings

The set of used technologies provides the ability to easily expand the system with additional elements. The same services, data access layer, and the same set of security can be used, for example, to build a website for mobile devices, enabling users register and purchase tickets online.

IV. DATA LAYER

The relational database PostgreSQL 8.4 has been used to build an application. The project database was created using pgAdmin III, which is part of the database installation package. The system consists of nine tables connected with primary and foreign keys.

Thanks to using object-relational mapping (ORM) business logic can be implemented based on the objects representation, this approach solves the problem of incompatibility of models (called impedance mismatch) [1]. The use of ORM increases the readability of code and minimizes the time devoted to writing long and complicated SQL queries. Java class are mapped to relational database tables based on the annotation describing the mapping of objects in the tables. The part of the mapped java class is shown in Fig. 2.

```
package domain
{
[Bindable]
        [RemoteClass(alias="ticketbooking.domain.Reservat
ion")]
public class Reservation
{
        public var id:Number;
        public var date:Date;
        public var position:MainElementPosition;
        public var calendar:MainElementCalendar;
public var description:String;
        public var username:Users;
        }
}
```

Fig. 2. Definition and mapping of the Reservation class in client application

Implementation of the domain model is a very important element of the system. It is used in many places during implementation of the functionality of the system. It is important that the implementation of this model was not related to other programming interfaces, and it has no influence on other tasks except for the business aspects [2].

Plain Old Java Objects (POJO) class are being used to implement the domain model of the system . Most POJO classes are working properly with Hibernate. It cause that Hibernate persistance API works the the most properly with business model implemented as POJO [2].

Domain model which is used in the server application is closely mapped to the model used in the client application written in Adobe Flex technology. BlazeDS application server and Flex allow for serialization of data between ActionScript language and Java. Implementation of the model in ActionScript and mapping it to Java class requires an addition of the RemoteClass tag to the class definition, specifying the package and the Java class, which the ActionScript class corresponds to. Example showing the definition and mapping of the Reservation class is shown in Fig. 3.

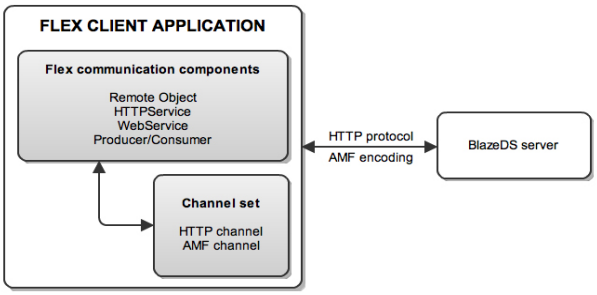

Fig. 3. Architecture of the client application using BlazeDS server

V. MVC ARCHITECTURE

MVC architecture (Table 1) was used in both parts of the application, the server part and client part. MVC architecture has the most important significance in the case of the client application, because there is a whole process of interaction between the user and application interface. The use of PureMVC framework provides a uniform method of controlling an application based on events (called event-driven) at every level.

Application data model consists of nine proxy objects. Almost every object from the domain model corresponds to the correct proxy object, in addition SecurityProxy object is responsible for maintaining security of the system. The main element of each proxy object are proxy responsible for cooperating with the part of the server [5]. These methods are only wrapper methods to cooperate with a specific remote service by using RemoteObject components.

Framework term view in Pure MVC is directly related to the mediator object. The system consists of several mediators, each responsible for a particular component of the user interface. All mediators shall cooperate among themselves and with other components using the PureMVC notifications sent through sendNotification method [5].

Command objects are corresponding to controllers which are the part of the PureMVC framework. In the described system command objects have two roles.
The first is the initial configuration of the framework, which includes registration of all mediators and proxy objects. The second one is the response to the notifications sent by the mediator-type objects and performing operations on proxy objects.

TABLE 1
MVC layer characteristics

Layer	Description
Model	Represents data, their logic and relations.
View	Responsible for displaying the data represented by the model.
Controller	Responsible for actions performed by the user and update the data represented by the model.

VI. COMMUNICATION LAYER

BlazeDS server provides highly scalable access to remote procedure calls service (RPC) and messaging services for client applications built in Adobe Flex technology. In other words, BlazeDS enables client applications to access data stored on a remote server and the exchange of messages between multiple clients connected to the server.

Communication layer consists of three types of Flex components: RemoteObject, Consumer and ChannelSet. Each of the RemoteObject components is connected to a server-side service. Consumer component and ChannelSet component are responsible for receiving messages sent from server to client application. Communication between the client application and BlazeDS server is shown in Fig. 4.

All communication between the Flex application and the BlazeDS server is based on the messages. Flex components use few types of messages to communicate with the corresponding server-side services. BlazeDS server uses two patterns of message:

- Request/reply pattern is used by RemoteObject, HTTPService, WebService Flex components. A component sends a request to a server and receives an answer.
- Published/subscribe pattern is used by the Producer and Consumer Flex components. Producer publish messages, and then the Consumer receives messages published by other customers.

Server-side application is a web application which runs on the Java EE application server. Requests from the client goes through the channel to the appropriate endpoint on the server. From the endpoint request goes through a series of Java objects such as MessageBroker, Service, Destination and Adapter [6]. When the request reach the last element it is supported by the appropriate java service.

VII. SECURITY LAYER

Security layer of the system was build thanks to the Spring Security package. This package is based on aspect-oriented programming. It cause that service layer of system can be design without thinking about security issue [1].
Application security is implemented at two levels.

– Securing a client application through the preparation of the application interface based on the roles of the user. Depending on roles of currently logged user, the various interface elements are shown and others are not shown.

– Securing access to services layer on the server-side application. This security is achieved thanks to annotations which are used in services declaration. The annotations describe which roles user has to poses to use specific service.

In addition, thanks to use of MessagingService and long polling technique available in BlazeDS server, application is equipped with an automatic logout process of the user whose account has been disabled by the administrator.

VIII. CONCLUSIONS

The main aim of this study was to show possibility of creating Rich Internet Applications based on Flex technology and how to combine them with the Java EE applications based on Spring framework. The process of creating an application was made in accordance with good practices aimed at reaching the goal of high quality computer system. Process consisted of the following stages: preparation of use cases, domain model and database, design of user interface, flow control and in the last stage implementation and testing of the system. In the result ticket booking system was created.

The resulting system is a multi-platform and can be constructed and implemented in two ways: as a desktop application which can be run using the Adobe AIR platform, or as a web application which can be run via a web browser. In addition, proposed solution caused that the system is very flexible and easily suitable for further development. The system is based on open-source technologies and it shows that it is possible to build a system which satisfies all the conditions of nowadays customers, without having to purchase expensive commercial licenses.

ACKNOWLEDGMENT

The authors are a scholarship holders of project entitled "Innovative education ..." supported by European Social Fund.

REFERENCES

[1] Rod Johnson, Juergen Hoeller, Alef Arendsen, Thomas Risberg, Colin Sampaleanu, *Spring Framework Profesjonalne Tworzenie Oprogramowania*, Helion, 2005

[2] Christian Bauer, Gavin King, *Hibernate w akcji*, Helion, 2007

[3] Deepak Alur, John Crupi, Dan Malks, *Core J2EE Wzorce projektowe*, Wydanie Drugie, Helion, 2004

[4] Jeff Tapper, Michael Labriola, Matthew Boles, James Talbot, *Adobe Flex 3 oficjalny podręcznik*, Helion, 2008

[5] Cliff Hall, *PureMVC Implementation Idioms and Best Practices*, http://www.puremvc.org, 03.02.2008

[6] *BlazeDS Developer Guide*, Adobe Systems Incorporated, http://livedocs.adobe.com, 2008

[7] Piero Fraternali, Gustavo Rossi, Fernando Sánchez-Figueroa, *Rich Internet Applications*, IEEE Internet Computing, vol. 14, no. 3, pp. 9-12, May/June 2010, doi:10.1109/MIC.2010.76

[8] J. Farrell, G.S. Nezlek, *Rich Internet Applications The Next Stage of Application Development*, Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on. 25/07/2007

[9] Dębiński A., Sakowicz B., Kamiński M.: "Methods of Creating Graphical Interfaces of Web Applications based on the Example of FLEX Framework", pp. 170-173; TCSET'2010, ISBN 978-966-553-875-2

[10] Janas, R.; Zabierowski, W.; "Brief overview of JEE"; Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2010 International Conference on; 2010 , Page(s): 174 - 176, ISBN 978-966-553-875-2

[11] Ritter R., Sakowicz B.: "Publishing and decisioning bidding system based on J2EE platform in combination with spring and hibernate technology", CADSM 2009, Ukraina, ISBN 978-966-2191-05-9

[12] Sakowicz B., Wojciechowski J., Dura. K. „Metody budowania wielowarstwowych aplikacji lokalnych i rozproszonych w oparciu o technologię Java 2 Enterprise Editon" Mikroelektronika i Informatyka, maj 2004, KTMiI P.Ł. , pp. 163-168, ISBN 83-919289-5-0