Web Services and Enterprise Games

COSTEL ALDEA[1], LIVIA SANGEORZAN[2], ALINA ALDEA[3]
[1]Department of Computer Science
[2]Department of Computer Science
[2]Library
University *Transilvania* of Brasov
Iuliu Maniu, 50, 500091, Brasov
ROMANIA
costel.aldea@gmail.com http://cs.unitbv.ro/~costel

Abstract: In the last years, the complex software solutions in bookkeeping domain have known a substantial growth on the market and involve the use of the newest technologies for developing safe, complex, flexible and according to the current legislation systems. The current paper presents a project which is designed to make bookkeeping operations using the computer and allows the obtaining of precise financial situation at any time. It is a user-friendly application and is familiarized with the needs of the market.
According to the accounts plan, the application will generate account books and periodical reports. The balance of trade can be edited so that the user can insert the values for debit and credit. The application also offers flexibility for the multitude of the economic indicators. There are default indicators that are used for the balance sheet or for the profit and loss report, but the user can define new indicators and can also edit their formula.
The project offers web services implemented in Java that can be used with any other applications and have as functionality the interpretation of any mathematical formula and the calculation of the two special indicators. The GUI was developed with Java Server Faces and MyFaces, building a friendly and easy to use structure.

Key-Words: - Web services; Distributed computing; Enterprise games

1 Introduction

This paper presents a web application which helps the trading companies to have a book keeping overview. The purpose of this project is to automatize the bookkeeping activities, to have an up-to-date standardized and electronic financial situation.

The increasing use of the software solutions on the economic plan leaded to the realization of the business resources administration systems, which start with simple programs for material planning and end with the administration of the production processes and of the financial activities. These systems are called ERP (Enterprise Resource Planning). They were created as a solution to the radically changes from the business domain, the growth of the company activities and the need to be permanently updated, giving the capacity of processing a high volume of data and information for the purpose of efficiency and optimization of the processes. Therefore, the bookkeeping modules are lately integrated in common platforms, such as ERP.

The project is based on the general accounts plan according to the current legislation. This can be adapted to the needs of the user, giving a flexible structure in which the user can add new accounts for its activities.

According to the accounts plan, the application will generate account books and periodical reports. The balance of trade can be edited so that the user can insert the values for debit and credit.

The application developed also offers flexibility for the multitude of the economic indicators. There are default indicators that are used for the balance sheet or for the profit and loss report, but the user can define new indicators and can also edit their formula. These types of indicators and their computing are meet in enterprise games theory.

Starting with the indicators, the user will be allowed to generate graphics to analyze the evolution of the selected indicators in the defined periods. There are two special economic indicators that can be analyzed and observed separately. They show the ability of the company to cover the debts.

Also, the periods of time for which the reports and account books are generated can be defined by the user.

The project also offers a centralized system for the administration of the users; this can only be done by administrators [7].

The project brings flexibility to the user to define new accounts and economic indicators and also the capability to generate graphics in order to have a general evolution overview.

The project is designed to be a web site [2] that uses web services for an accountancy application. It is structured in two parts: web services implementation and

web service client which is the main application. Both parts are implemented in Java [1].

2 Analyze and design

Figure 1 illustrates the whole scenario of the application.

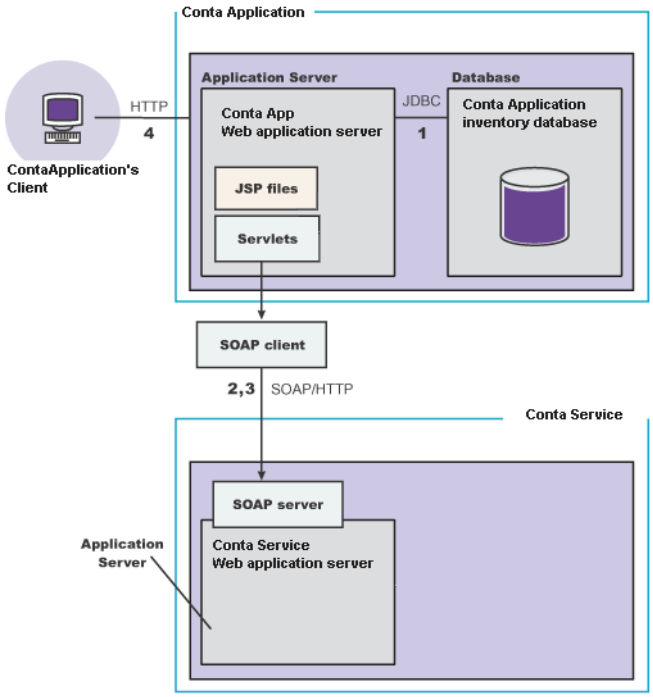

Fig. 1 Project design

The web services are independent from the client part of the application. It can be included or used also with other applications. There are three web services: for calculating of any formula after receiving it's parameters with the help of a formula interpreter, and the other two web services will calculate special economic indicators based on their formula and receiving their parameters.

Web service client is an application used for accountancy. It is a multi-user application. Its requirements are to administrate and control the general plan of accounts, define and manage account, define and manipulate economic indicators, draw graphics, calculate special indicators and generate periodic reports like "balance sheet" and "profit and loss".

The following events happen when a client (in this case should be a user) on the web site [4] wants to calculate the values for the *LC Indicator*- an economic indicator:

1) ContaApplication invokes the "calculate method" which is bound to Conta Service.
2) ContaApplication invokes a SOAP client that communicates with a SOAP server at the ContaService sending also the parameters taken from the database used in the formula of *LC Indicator*.
3) Conta Application obtains resulted values for *LC Indicator*.
4) Conta Application sends the results to the client.

The application was implemented with Java Server Faces.

2.1 Patterns

The use patterns are a ubiquitous way to abstract a problem and its solutions. Because patterns are recognized by all developers and architects, patterns can save time and energy. A pattern is a proven solution to a well-known problem. You can reuse patterns, and this reuse helps solutions become robust.

2.2 Java Server Faces

JSF architecture is a framework for Web applications. It is driven by the Java Community Process (JCP), and is expected to become a standard framework for Web application developers. At present, the more than 50 frameworks for developing Web applications indicate a strong need to standardize the framework and JSF architecture is doing that job.

2.3 Model-View-Controller (MVC)

The MVC pattern's purpose is to decouple Model (or data) from the presentation of the data (View). If the application has more than one presentation, it can replace only the view layer and reuse code for the controller and model. Similarly, if it needs to change a model, the view layer remains largely unaffected. Controller handles user actions that might result in changes in the model and updates to the views. When a user requests a JSF page, the request goes to FacesServlet. FacesServlet is the front controller servlet used in JSF. Like many other Web application frameworks, JSF uses the MVC pattern to decouple the view and the model. To handle user requests centrally, the controller servlet makes changes to the model and navigates users to views.

FacesServlet is the controller element in the JSF framework which all user requests go through. FacesServlet examines user requests and calls various actions on the model using managed beans. Backing, or managed, beans are an example of the model. JSF user interface (UI) components represent the view layer. The MVC pattern helps to divide tasks among developers who have different skill sets so tasks can be carried out in parallel; that is, GUI designers can create JSF pages with rich UI components while back-end developers can create managed beans to write business-logic specific code.

2.4 Web services

A web service is a piece of business logic, located somewhere on the Internet, that is accessible through standard-based Internet protocols such as HTTP or SMTP. Using a web service could be as simple as logging into a site or as complex as facilitating a multiorganization business negotiation.

Given this definition, several technologies used in recent years could have been classified as web service technology, but were not. These technologies include win32 technologies, J2EE, CORBA, and CGI scripting. The major difference between these technologies and the new breed of technology that are labeled as web services is their standardization. This new breed of technology is based on standardized XML (as opposed to a proprietary binary standard) and supported globally by most major technology firms. XML provides a language-neutral way for representing data, and the global corporate support ensures that every major new software technology will have a web services strategy within the next couple years. When combined, the software integration and interoperability possibilities for software programs leveraging the web services model are staggering.

Web services provide a layer of abstraction above existing software systems, such as application servers, CORBA, .NET servers, messaging, and packaged applications. Web services work at a level of abstraction similar to the Internet and are capable of bridging any operating system [6], hardware platform, or programming language, just as the Web is.

Unlike existing distributed computing systems [3], Web services are adapted to the Web. The default network protocol is HTTP. Most existing distributed computing technologies [5] include the communications protocol as part of their scope. With Web services, the communications protocol is already there, in the worldwide Web.

New applications become possible when everything is Web service enabled. Once the world becomes Web service enabled, all kinds of new business paradigms, discussion groups, interactive forums, and publishing models will emerge to take advantage of this new capability. Software and hardware vendors alike are rushing Web services products to market. The widespread adoption of the core standards represents a significant breakthrough in the industry. Applications can truly be built using a combination of components from multiple suppliers. Specialists are emerging to provide services in the areas of security, transaction coordination, bill processing, language translation, document transformation, registries and repositories, accounting, reporting, and specialized calculation. Applications being built anywhere, anytime, on any system can take advantage of prebuilt components, speeding time to market and reducing cost.

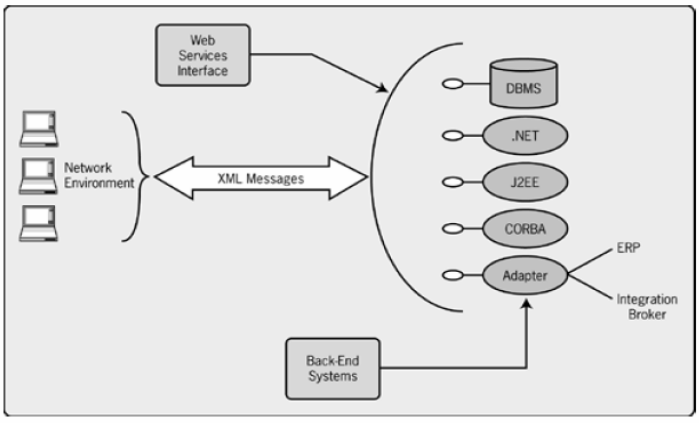

Fig. 2 Web services interface with back-end systems

Web services are Extensible Markup Language (XML) applications mapped to programs, objects, or databases or to comprehensive business functions. Using an XML document created in the form of a message, a program sends a request to a Web service across the network, and, optional receives a reply, also in the form of an XML document. Web services standards define the format of the message, specify the interface to which a message is sent, describe conventions for mapping the contents of the message into and out of the programs implementing the service, and define mechanisms to publish and to discover Web services interfaces (fig. 2).

3 Implementation

ContaApplication as well as Conta Service were implemented with Netbeans IDE 5.0.

In NetBeans IDE 5.0, the New Project wizard has been extended so that while you create a web application, you can specify that you want to use JSF, Struts, or both. When you make this selection, the IDE adds all the JSF and Struts libraries to your application, as well as all their configuration files. While using the configuration files, you have code completion to support you as well as the ability to have menu items create all the tags the configuration files need. The New File wizard has also been enhanced - one can choose templates for the creation of JSF Managed Beans, Struts Actions, and Struts Form Beans.

All the libraries needed to create and deploy a web service client are bundled with the IDE, so that the web service client created in a J2SE application can be deployed without a problem. Also, in web applications, you can now call web service operations directly from a JSP page, so that you don't even need to create a servlet anymore. Finally, a new wizard has been added - it can now create a WSDL file directly in the IDE and use it to generate your web service files.

3.1 Netbeans APIs for Web Services

Historically multiple APIs were created in Netbeans to facilitate and unify the Web Services support in multiple project types. Initially in Netbeans 5.0 the support was created over the JAX-RPC architecture.
The JAX-RPC service creation is supported in:

- Web Application (J2EE 1.4 project) ;
- EJB Project (J2EE 1.4 project).

3.2 Java API for XML-Based RPC

For typical Web service scenarios, using JAX-RPC reduces complexity for developers by:

- Standardizing the creation of SOAP requests and responses;
- Standardizing marshalling and unmarshalling of parameters and other runtime and deployment-specific details;
- Removing these SOAP creation and marshalling/unmarshalling tasks from a developer's responsibilities by providing these functions in a library or a tool;
- Providing standardized support for different mapping scenarios, including XML to Java, Java to XML, WSDL-to-Java, and Java-to-WSDL mappings.

JavaTM API for XML-based RPC (JAX-RPC) supports XML-based RPC for Java and J2EE platforms. It enables a traditional client-server remote procedure call (RPC) mechanism using an XML-based protocol. JAX-RPC enables Java technology developers to develop SOAP-based interoperable and portable Web services. Developers use the JAX-RPC programming model to develop SOAP-based Web service endpoints, along with their corresponding WSDL descriptions, and clients. A JAX-RPC-based Web service implementation can interact with clients that are not based on Java. Similarly, a JAX-RPC-based client can interact with a non-Java-based Web service implementation.

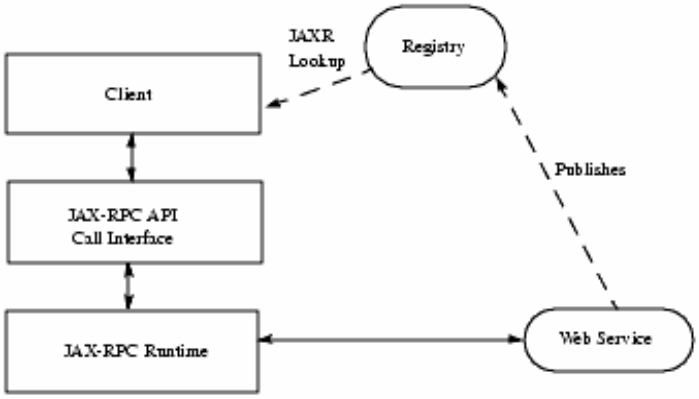

Fig. 3 JAX-RPC

For typical web service scenarios, using JAX-RPC (fig. 3) reduces complexity for developers.

3.3 Develop the Web Service

The next steps describe how one develops the web service using NetBeans. These steps show how to create a new project for the service and how to create the web service itself, which means writing the implementation code and–optionally–add a SOAP message handler for logging purposes.

3.3.1 Create the Web Service

The New Web Service wizard helps the user to create the skeleton for the web service. When the user exposes the service from a web application, the wizard creates two Java source files: an implementation bean class and a service endpoint interface. The endpoint interface lists the web service methods that you want exposed, while the implementation bean contains the actual implementations for the different service methods. Since all service methods or operations require the java.rmi.RemoteException exception, the wizard includes this exception for you.

3.3.2 Add operations or methods to the web service

One use the Add Operation dialog to define: the name of each method or operation, its return type, the required parameters, and exceptions other than java.rmi.RemoteException. In the Name field one enter the method or operation name. It set the operation return type either by selecting a type in the Return Type drop–down list or entering a type. In the Input Parameters and Exceptions section, it adds and defines all the parameters and application–specific exceptions for this operation. It clicks Add for each parameter and exception you want to define. For parameters, indicate the type of the parameter and provide a name. It can also mark a parameter as Final. For exceptions, it needs only to enter the type of the exception. The IDE adds the method signatures to the implementation and endpoint interface Java files.

The service endpoint interface file includes the declaration of every operation exposed by the web service. In ContaService the generated FormulaSEI.java contains the declaration of the opFormula operation just defined (shown here with comments removed):

```
package services;
public interface FormulaSEI extends java.rmi.Remote {
public java.lang.String opFormula(String[] var, String []
valori, String formula) throws java.rmi.RemoteException;
}
```

The implementation file contains the same operation declarations as the endpoint interface file at this point.

3.3.3 Create and configure SOAP Message Handler

A SOAP message handler intercepts request and response communications between your application and the web service, and it is tied to the web service endpoint. A message handler can provide additional processing to the message, such as encryption and decryption, service access control, transaction management, auditing, logging, metric collection, and so forth.

4 Database

The database for this project was implemented with MySql. The MySQL software delivers a very fast, multi-threaded, multi-user, and robust SQL (Structured Query Language) database server. MySQL Server is intended for mission-critical, heavy-load production systems as well as for embedding into mass-deployed software. MySQL is a registered trademark of MySQL AB.

The technology that stands at the basis of binding an SQL-database and a Java application is called JDBC.

4.1 JDBC

The Java Database Connectivity (JDBC) API is the industry standard for database-independent connectivity between the Java programming language and a wide range of databases – SQL databases and other tabular data sources, such as spreadsheets or flat files. The JDBC API provides a call-level API for SQL-based database access. JDBC technology allows you to use the Java programming language to exploit "Write Once, Run Anywhere" capabilities for applications that require access to enterprise data.

The JDBC API makes it possible to do three things:

- Establish a connection with a database or access any tabular data source;
- Send SQL statements;
- Process the results.

As a core part of the Java 2 Platform, the JDBC API is available anywhere that the platform is. This means that your applications can truly write database applications once and access data anywhere. The JDBC API is included in the Java 2 Platform, Standard Edition (J2SE) and the Java 2 Platform, Enterprise Edition (J2EE), providing server-side functionality for industrial strength scalability.

4.2 Using JDBC

If one wants to use JDBC, the first thing to do is to import the packages or classes that will be used in the class. The java.sql package (the JDBC API), is made available when the following line of code precedes the class definition: *import java.sql.*;*

4.3 Database Relationship Model

The database for this project is concentrated (fig. 4) on four important entities: user, indicator, account and period.

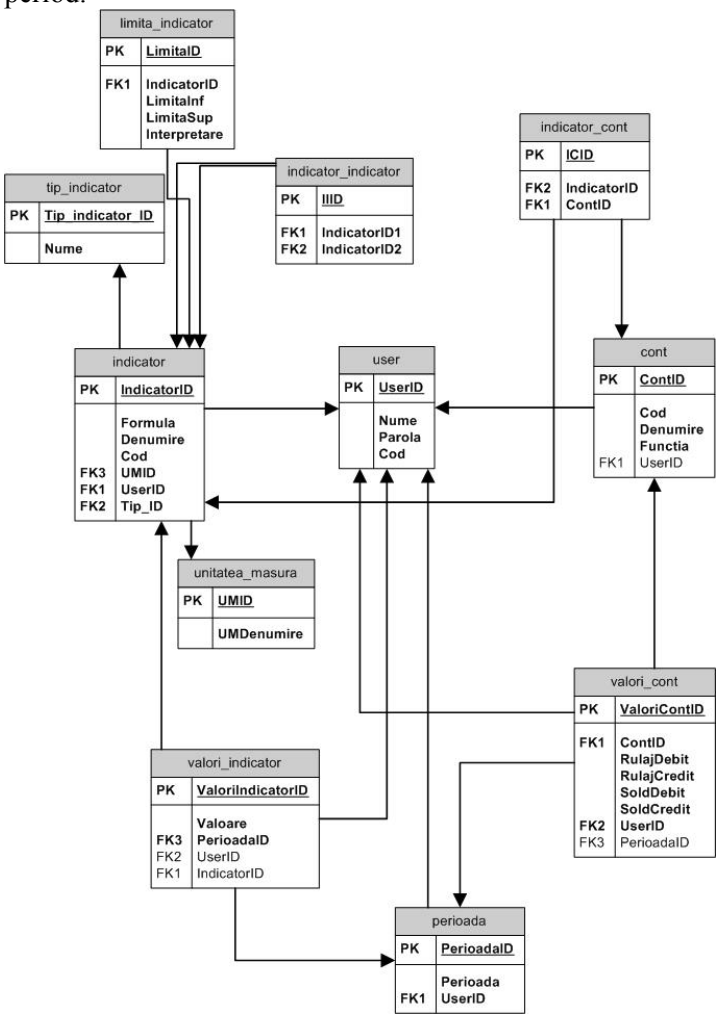

Fig. 4 Entity Relationship Model

The *user* is the one who will login to the project and has the power to modify, add and remove the most of entities. A user can be a simple user or can be an administrator. An administrator has the capability to make the changes to general data's and to the attributes of the rest of the users. A simple user can modify just its self defined data's but has the access to the general data's too. The user can define its own accounts, periods and indicators, that's why the rest of these entities are depending on the user table through a foreign key. Users are part of administrative section of the project

Account entity is the main entity for the accountancy-logic of the project. The account's attributes are: code, name and function. Each account has its owner user. The values for each account are memorized in "account-values" table. Here are recorded the account's values depending on period and user. This means if an account is created from an administrator it will have values depending on each user and each period of the user.

Else, if it is defined from a simple user the values will be populated just for this user depending on user's defined periods.

Indicator entity has as attributes: code, name, formula. The UMID is a foreign key for "unit of measure" table. The unit of measure values is predefined for this project: RON, % and days. An indicator has also a type, it can be a normal indicator or one used in the "balance sheet" or "profit and loss" form. The owner user of this indicator is recorded through the UserID foreign key. The values of the indicator are memorized in "indicator-values" table. It is the same concept like the "account-values" table. One indicator can have a range of limits that are describing between which bounds its values should be enclosed. "Formula" of one indicator is depending on others indicators or accounts. "Indicator-indicator" table shows which indicator depends on others and "Indicator-account" describes the relationship between an indicator and accounts.

5 Conclusion

By using the distributed technologies the projects creates a bookkeeping application, designed to the use of trading companies.

It has multiple functionalities for accountancy: defining accounts and indicators, generating graphics, account books and reports.

Some other enterprise games can use the web services offered by the project. The project offers some indicators as web services due to their complexity and to make them transparent to the clients.

References:

[1] Sangeorzan L., Aldea C., Dumitru M., *Java. Aplicatii*, Infomarket Publishing House, Brasov, 2001.

[2] Sangeorzan L., Aldea C., *Tehnologii Internet*, University *Transilvania* Publishing House, Brasov, 2003.

[3] C. Aldea, F. M. Boian , *On evaluating the performance parameters in a distributed system*, Proceedings of the International Conference on Knowledge Engineering, Principles and Techniques, KEPT2009 Cluj-Napoca (Romania), July 2-4, 2009, pp. 25-28.

[4] L. Sângeorzan, E. Helerea, C. Aldea et. al., *Aspects Regarding the Quality of Education Processes Based on WEB 2.0 Technologies*, Proceedings of the 12th WSEAS International Conference "Applied Informatics and Communications", 20 -22 August, Grecia, 2008, pp. 499-505.

[5] Boian F.M., Programare distribuit în Internet. Metode si aplicatii, Editura Albastra, Cluj, 1998.

[6] Boian F.M. si altii, Programare concurenta pe platforme Unix, Windows, Java, Editura Albastra , Cluj, 2002.

[7] L. Sângeorzan, Tehnologii web şi WebDesign, Ed. Univ. Transilvania din Braşov , 2009.