Architecture of web client server systems with advanced asynchronous communications

FILIP MALY, ANTONIN SLABY
Department of Informatics and Quantitative Methods
University of Hradec Kralove
Rokitanskeho 62, Hradec Kralove, 500 02
CZECH REPUBLIC
filip.maly@uhk.cz
antonin.slaby@uhk.cz

Abstract: - In the paper there are outlined results of solution of important and difficult problem of solution ways of communications in special client server systems having hybrid and from many points of view restricted client. There are described basic improvements of client enabling performance of more advanced software processes. In addition to it there is shown common concept and architecture of web client server systems that use asynchronous communication through one always open port (port No. 80). These systems have wide use and variety of non trivial application possibility.

Key-Words: - client server systems, asynchronous communication, software visualization, motivation, hybrid client, software processes.

1 Introduction
Basic purpose of this article is to describe common concept of web client server systems that use asynchronous communication through one port (port No. 80) and outline their architecture. The authors of this article tested four variants of solution of communication among parts of such systems. These variants included communication through several ports, communication through two ports which does not require permanent connection, one port communication having permanent connection access to server and finally one port communication that make use only temporary (on demand) server connection.

The process of testing and assessing various kinds of implementation resulted in reduction of number of communication ports so as the system could be used also in environments having strongly restrictive internet access policy. The design of such systems is rather easy unless there is not required implementation of processes that require asynchronous data access and exchange emanating from both client and server site of the system.

2 Software used for making elements of web client server systems with asynchronous communication of
Basic elements of the system that enable asynchronous communication web client with server include Client which is in our case written as applet, server part of the system which consists of so called servlets. Third element of the system is some library which is in our case set of web pages. These pages may set up asynchronous client response as a result of interaction with user of the system In tested samples of architecture was used Java script code for calling methods of client applet.

3 Software client, its features and ways of modification of client part of the system and its implementation
It is necessary to define and implement special interface able to accept requirements originating from web pages through Java script. It is necessary to suppose that hybrid client will respond to several kinds of requirements and should be able to cope with more than one requirement. This problem can be solved by introducing a collection of so called listeners which are able to elaborate events rising in the environment of HTML pages with help of JavaScript. In such a way are spread possibility of applet event handling and elaborating by other GUI elements that are outside applet. We suppose mainly two ways of implementation of listeners:
- Delegation of responsibility for event elaboration to server using applet
- Direct elaboration of the event by the client side

In the first case basic implementation of delegating listener may be given. In the second case we deal with the certain implementation of listener interface. Let us call it JavaScriptListener.
In case we consider and implement JavaScript functionality on the client side (web browser side) it is suitable to define simple package of listeners in java programming language able to elaborate the events by client side of the software. In addition to this, it is suitable to create the library JavaScript pieces of code looking after transferring events of browser to applet (hybrid client). The following sample of the code shows interface that defines heads of basic methods representing listeners. They are in fact methods implemented by client.

```java
public interface JavaScriptListenerInt {
    public void requestExample(String clientId, String strExample);
    public void requestExample(String clientId, Object objExample);
    public void requestExample(String clientId, InputStream isExample);
    public void commandServer(String clientId, String strCommand);
    public void commandServer(String clientId, String strCommand,String strData);
    public void commandServer(String clientId, Object objCommand);
    public void commandServer(String clientId, Object objCommand,Object objData);
    public void commandServer(String clientId, InputStream isCommand);
    public void commandClient(String clientId, String strCommand);
    public void commandClient(String clientId, String strCommand,String strData);
    public void commandClient(String clientId, Object objCommand);
    public void commandClient(String clientId, Object objCommand,Object objData);
    public void commandClient(String clientId, InputStream isCommand);
}
```

These methods can be called through JavaScript. They can be divided into three groups.

- **Methods that hand in commands (tasks) to client only.** These methods serve the purpose of change the state of client or its graphic user environment. These methods can be in service of combined request for client and server at the same time.

- **Methods that hand in commands to server only.** These methods are of course also handed in through client as there is not possible/adviseable to address server directly by JavaScript, as it might be safety hazard.

- **Methods that contain various requirement coming from web pages.** In spite of the fact that they may be included into one of the two previous groups, they are dealt with separately, as they serve for solution of completely specific requirements.

Software implementation of above mentioned methods successfully uses so called overloaded variants of methods, i.e. methods having the same name (identifier) but differs in number and type of parameters. Our implementation uses String, Object and InputStream types of parameters.

### 4 Elements of architecture of the system

Let us describe and comment proposal of common software architecture of the system from the implementation point of view. Next figure shows the basic concept of architecture.

![Basic concept of architecture](image_url)

The crucial task of the system is correct elaboration of response of client and server to events delivered by JavaScript. The result of such responses may cover up for example changes of content of selected window of the browser, some change affecting client only, server only or both client and server, various combinations of these tasks and many other responses. Due to these reasons the response is executed individually in the process of applications of various listeners. It is but also possible to assume partial solution which is analogous to that by listeners in applets, but is implemented on browser and Through Java Script. Above mentioned solution offers the ideal possibility of binding applets with web pages in accordance with intentions and ideas introduced during time period when Java and JavaScript were
invented and started to spread. Java Script that covers completely the whole area of processing events for both hybrid client and server side is necessary presumption for establishing common concept of the system capable to integrate more complicated tasks into the environment of web browser. Making use of Java script in the role of communication mean between applet and web page is very reasonable and may be enriched by some attitudes used in AJAX technology etc.

5 Way of communication between/among elements of the system
Important step towards creating of common architecture is to set up (define) common protocol for communication among various parts (modules) of the system. In our system is for this purpose established special class Protocol. The communication inside the system is performed by special objects. These objects also serve for data exchange.

The main characteristics of the class Protocol. The system has to implement special interface as the class must have ability of mobility among various parts of the system so that its descendents could be serialized.

The class possesses special static method for verification of special marks of the protocol. It also implements methods for basic manipulation with specialized object HashMap that serves for information entry.

6 Object HashMap
Class HashMap proved to be very useful possibility and good help for solving our protocol problem. Class HashMap organization is similar to that of array. Classical array is indexed with indexes of type integer. HashMap but is indexed by so called key, which can be some object. In our case was for the purpose of indexing used type string. String keys of course have to be unique. The advantage of this solution is in the fact, that in such a way we can include into HashMap some data or information and keys possess also role of meta information of some kind. Consequently keys may be used for identification of elements in the HashMap and so into the HashMap object can be inserted almost any kind of information. Keys must be defined in the scope of protocol so that we could recognize what was inserted into the Hash map. There are two possibilities for making key definitions.

- All key are defined in predecessor of Protocol object.
- Keys are defined in ancestor objects of Protocol object. More information on this object will be given in extended version of this article.

In case we want to design sufficiently common and widely useful architecture we have use special mechanism. One possibility is to use systematically the class Protocol, which is ancestor of all protocol classes. The class instance can be retyped only in the moment, when it needs to obtain data. This is the only possibility to ensure that the architecture is as much independent on concrete data exchanged between modules of the system as possible.

7 Basic requirements placed on the architecture and essentials of its functionality
We will describe requirements and principles laid on the architecture so that it could be used for various purposes in this part of the text. Basic requirements placed on the architecture:

- Possibility to send requirements to the server with the help of web browser client.
- Possibility to execute such requirement on the server i.e. possibility to execute concrete processes on server. The processes may translate/interpret source code of programming language, establish various calculations, create database table, and do many other tasks.
- Possibility to send back and show results of these processes executed on the web browser, on the client or on both. (It depends on specific requirements placed on the system and on specific situation).
- Possibility to create easily simple testing examples.

The architecture is based on the following essential facts/principles:

- The client transforms data to the form of Protocol.
- Server can accept everything which is instance of the class Protocol.
- Server can run everything which is instance of the class Requirement. Instance saved in the variable Requirement specifies what is to be done on the server with data (translation/interpretation calculation etc.).
- Client accepts and elaborates data in case they are in instance of the class Protocol.
- Client runs, redirects etc. obtained data according to its function.

The new functionality of the system is established relatively simply by introducing the new client implementing extensional purpose.
8 CASE study

Described component enabled efficient education of specialized subject in learning management systems (LMS). The main purpose of just described system was to add to LMS the new functionality that could run (execute) Java environment and other software packages from the environment of LMS in such a way that students are not aware of difference between running test programmes from the inside LMS and running it in Java environment and/or other native software packages.

The results obtained.

The software described in the article proved to be fully capable of integration into LMS. Software system was capable to be executed from LMS. It is also able to run without dependence on LMS. System is easily accessible from any place in the internet through web browser and web interface. System is easily capable to run any set of test examples and respect all standards and environment details students got used during direct work with Java environment and/or other software packages. The installation and configuration requirements revealed to be very minor. Working with the system was without problems. The testing was carried out at the Faculty of Informatics of Management of the University of Hradec Kralove during distance education subject Object oriented programming with Java. Java environment was addressed and executed from LMS Moodle and all features and functionalities proved to be without problems. Other testing included running software package Matlab from LMS again without problems. System was partially and successfully tested also with rather restricted mobile clients.

9 Conclusion

The system described in the article proved to be very flexible, easy to modify. In such a way it promises wide range of applications. Other details of the architecture and more applications will be given in extended version of this article. Various variants of the system contain some core ideas that can serve as design patterns in the area of asynchronous communication. Other applications are prepared and tested.

This contribution is supported by the grant project No 402/08/1046 of the Czech grant agency (GACR): Models of firms having mobile architecture.

References: