Secure Connection of Database Systems to the Applications made by a Standard MVC

Liberios Vokorokos, Ondrej Látka, Martin Chovanec, Alžbeta Kleinová

Department of Computers and Informatics, Technical university of Košice
Letná 9, 042 00 Košice, Slovakia
E-mail: liberios.vokorokos@tuke.sk, ondrej.latka@tuke.sk,
martin.chovanec@tuke.sk, alžbeta.kleinova@tuke.sk

Abstract: The development of information technologies in last years reaches swift increase. All of the systems involved to information process must react to this increase. The main medium in these surroundings is information, which value is still increasing. The creation of applications must flexibly react to development, but also to importance of information, and their safe transfer, transformation and safe storage. Current trend in creation of these applications is their functional dividing into the several layers. The one of the standards that creates these applications is also MVC i.e. standard Model-View-Controller, which belongs to three-ply applications. This paper describes the interconnection of three-ply applications in internet network to database systems, security components in these applications and their most effective combinations. The contribution is a concrete design realization of the connection to the database, and design of its most effective security combinations considering the standard MVC.

Keywords: design patterns, MVC, Model-View-Controller, servlet, JSP, application patterns, https, secure of database, secure storage of information.

1 Introduction

In time of swift internet development, the big demand for personal access applications springs up, where identity of client is monitored and analyzed. For those applications are not sufficient the classic static pages, that have just informatively character, what was the reason for creation of dynamically generated pages principles. However, these principles require transformation and formation of new implementations, which are divided by the methods of source code generating. The quest is to join all of those implementations and to make one standard, which would suit all requirements of functionality and performance. Those requirements are influenced by different factors, between which belong also principles of connection to the database and implementation of secure modules. In

the three-ply applications are used specialized database connectors, which form data stream between application and database. For this data stream is necessary to use secure encrypted channel. After the safe receiving of transmitted data, they have to be stored securely, mainly those, which are sensitive to misuse. In Department of Computers and Informatics is this problem researched, in cause to expand e-Learning in university. It is also a research item of VEGA 1/1064/04 project, which deals with distributed systems, that use computer networks, and also security of transmitted data and their suitable storage.

2 Principles of Multilayer Standards

2.1 Introduction to Multilayer Applications

Distributed network applications belong to multilayer applications. Those applications have minimally two layers. First is interaction layer which procures communication with client and the second is processing layer which procures processing requests from client. Interaction layer offers interface for client and receives the coming requests, which are transformed and sent to the second layer for processing. The interaction layer acquires results from the processing layer, transforms them and sends them back to the client as a response to his request. The second layer processes the requests which are coming from the interaction layer with the assistance of implemented "business logic". This layer can include an access to the database or the cooperation with other EIS systems (ERP, or other back-end systems). Standard Model-View-Controller (MVC) is derived from this principle. MVC design pattern presents an implementation of multilayer architecture for network applications. MVC creates applications by principle of dynamically generated pages. These pages are generated or created in the application runtime, namely directly according to the client's requests. So that these applications can work, they must start on specialized servers up, which process client's requests and later on generate source code. The most often used technology for creating these applications is JAVA. This technology helps to divide working logic of the application better. However, the term Model-View-Controller was never exactly defined, its importance and big meaning is acknowledged only in present. The exact definition of the term Model-View-Controller is still in dispute. The basic idea of multilayer application is to divide the application into multiple layers, and thus separate the responsibility for the appearance, the behaviour and the control of the application. By this decomposition of the applications into the single parts or layers, it is bound to ensure a sufficient communication between layers.

2.2 Cooperation of Components MVC

The model presents the processing layer, which defines business logic of application (i.e. calculation, making of contracts, export of contracts etc.) and runs database command, like in Figure 1. The model keeps data and relationships between them without aspect on how the data will be presented. It means, that presentation can change in dependence on the type of client (web client, swing client etc.), but model remains the same. It can differentiate in dependence on the type of architectures, which is used by the application. In two-ply applications, where the web layer is in direct interaction with data stored e.g. in database, the set of ordinary Java objects can be the classes of model. These objects can be created manually with the assistance of the data's set *ResultSet*, which is a result of database command. The other option is to create an instance, and they can be filled automatically with ORM frames – object-relational mapping (support frames e.g. TopLink or CocoBase). In the more complex company's applications (where layer communicates e.g. with EJB) Model is created with Enterprise JavaBeans. This access is still some more exacting in the systems devices, because if web layer is trying to use entity *beans* directly as a part of model, the number of remote calls will increase. In majority of the cases these beans are coming back to web layer, where they are used for creating of the dynamically contents. Also, they are called data transfer objects, or value objects. All of the processed objects are sent to the view layer, which have the task of interpreting the data.

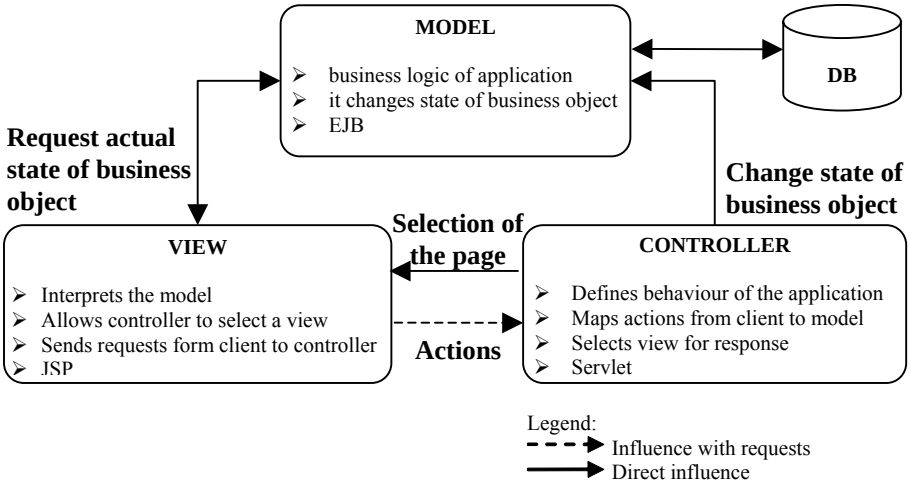

Figure 1

Communication of layers in standard MVC

View is layer, in which the graphic part of application is created. The majority of dynamically parts of page are generated in this layer. Some applications require JavaScript on client site, what is not violating or exceeding the conception of standard MVC. HTML and JSP are not the only one option for creating the view.

The WML can be used instead of HTML. Because the view is separate from model, it can support several views (each for different client type) and in the process use the same model's components. These two layers are connected through the other layer that is called controller. For each client's action the controller associates the functions implemented in model business logic, what is demonstrated in Figure 1. After receiving the response from client, the controller retransforms it to the concrete operation. Then controller processes this operation, or delegate part, which will process the operation. After this operation, controller cooperates with view layer to select concrete page view, which is later on sent back to client. The controller centralize logic for redirection requests from actual view to another view on the basis of coming request's URL, parameters of request and state of application. Controller helps to separate presentation's components (views) from internal operations. This property increases security of created applications, because individual view can see only information, which is sent to them. In other case, the views can not influence the control elements and also they do not have the possibility to influence the stored information. In applications as those, which are based on principles MVC, the controller is a servlet that is common point, across which gets all of the requests. In controller is possible to implement functionality like authorization, authentication, logging and all of the similar safety elements, which design is described in next chapter. In this architecture are JSP pages used just for presentation, the application's logic is obtained by a servlet (servlets). The development and maintenance of the application based on standard Model-View-Controller is the easiest, because all parts have exactly defined tasks.

3 The Security Suggestion in the MVC Applications

In quest of securing the information against the misusage or the loss, it is necessary to deal with questions as: where will be the application used, where will be the information stored, how will it be relayed etc. Therefore, it is needful to pay attention to each communication channel in order to sufficiently secure it, even though the channel is not used. The browser participates in the information transmission, and communicates with the application through the view layer. The information flow proceeds through the controller, which communicates with the model that extends connection with the database. In this way one big communication channel is made, which can be divided into the several parts. The transmission between the browser and the application, and the transmission between the application and the database system are very crucial.

The Communication between the Application and the Database

The reason why it is necessary to secure also such a connection is that in the case the big database systems with the high number of records are used, these database

systems are located in the individual servers, which can physically be in the different places. By the safety experiments to secure such a channel was the *Stunel* used, that forms encrypted channel. Tunnel is defined as two end nodes, namely input and output from the tunnel and as a packet transmission mechanism in the tunnel. For the input and output from the tunnel is the message encrypting used and the transport mechanism is used for packet security. This mechanism encases the safety packet (inner) into the simple packet (outer), so the original packet stays unreadable. The packet security is made on the seventh layer of the OSI model, through which the packet is proceeding in the constituent layers up to the client that decodes the packet the same way. This principle is described in the Figure 2. This technology is making an effort to create safe transmission channel, which in the case of connection failure disrupts any communication with the surroundings. This technology creates secure communication protocol over the TCP protocol and to secure the packets it uses the combination of encryption by public and private key. Asymmetric encryption is used in the first phase, i.e. the authentication and the generation of the keys, and private key is than used for the message encryption. Various algorithms are used for the message encryption. Within the task solution at the Department of Computers and Informatics, of the Faculty of Electrical Engineering and Informatics was used the MD5 hash function with the 128 bites cryptography, and that not only for the transmission encryption but also for the data encryption, which are recording in the database and are exclusively appointed for the client.

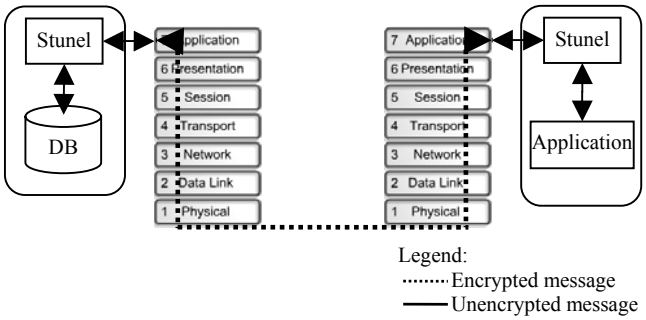

Figure 2

Information transmission with assistance Stunel

The Communication between the Browser and the Application

The necessary reason for information transmission security from the browser to the application is data protection against the unauthorized access. The standard HTTP protocol does not support any relayed data protection. Therefore, if it is necessary to secure this communication, the encrypted variant HTTPS has to be used. In this context the abbreviation SSL (Secure Socket Layer) is used, which is additional encrypting layer that creates secured channel under the standard HTTP protocol. The power of the encryption also depends on the encrypting key length,

which nowadays uses 128 or more bites encrypting key, which is relatively safe. Since, the information are transferred up to the model layer, the design pattern has to support this security protocol, otherwise the data will not be processible and the model will not recognize the request from the view layer. HTTPS is supported by the majority of the internet browsers, and also the majority of the design patterns.

4 The Implementation of the MVC Applications with the Databases

In the last chapter the security model of the MVC standard by the different type of communication was described. The one of the communications was also the communication with the database. In order to set in this communication it is needful to create the connection with the database. This Chapter deals exactly with this problem. The distributed network applications, which are created as dynamic web pages, necessarily need connection to the database in order to gain from its sources, and also to record the data. In the research at the Department of Computers and Informatics, of the Faculty of Electrical Engineering and Informatics was used the MySQL database system and as a proposal language JAVA. The application made by MVC standard should be sufficiently adaptable and flexible, and the most importantly the application should be safe enough [5], [6]. This connection operates in various ways, namely for different design patterns and for different database systems. The first phase is to build the connection with the database. It is followed through the database connector, which is freely accessible for each database system. In order that the application will know how to use this database connector, it has to be configured in advance and later on the processing layer has to build the connection with the database connector. This connection can be created in two different ways. The first way is to do it manually, where the database connector configuration and subsequent connection are formed with the assistance of one instance. In this instance the access paths to the connector and the individual parameters of the connecting database are set [8]. Such a manual connection is displayed on the Figure 3.

```
private MySqlConnection() {
    try   {
          Class.forName ("com.mysql.jdbc.Driver", "app/drivers/mysql");
          con = DriverManager.getConnection(dbURL, "root","root");
               } catch (SQLException e) {
                  e.printStackTrace();
               } catch (ClassNotFoundException e) {
                  e.printStackTrace();}
          }
```

Figure 3

The proposal of MySQL database connector setting

By this connection is needful to secure that such a database connector registration and connection with the database will be formed only once during the whole application running [3]. In converse case, after every request the connection will be created until the system will be destroyed, what could result in the disruption of the stored information consistence. Therefore the new instance is usually formed, which creates the connection, and this instance is controlled by each class call, whether the instance exists or does not. If the instance does not exist, new instance is formed. The contribution of this suggestion is that the creating class can be located in one file, and so it can be simply controlled and the information about the connection can be kept, as it was described before. In this way the instance is simply created, and will not be called in double entry. The second way of creating the connection and setting the database connector is by using surroundings defined by proposal framework, which from the configured data determines, where is the connector located, what are the access data to the database system, where is the setting database started etc. In this case the processing layer, which is the model, probes in the configuration file the information about the connector, the database, and by itself builds a connection [1], [2], [7]. After this operation, the required information and also the information, if the connection was built, are put into the helping buffer. After every another attempt on the connection the model will know, that the connection is built and will not be trying to form another connection.

```
<data-sources>
   <set-property
     property="autoCommit"
     value="false"/>
  <set-property
     property ="driverClass"
     value=" com.mysql.jdbc.Driver
     path="app/drivers/mysql"/>
   <set-property
     property="maxCount"   value="4"/>
   <set-property
     property="password"  value="root"/>
   <set-property
     property="url"
     value=" jdbc:mysql://localhost:3306/DatabaseName"/>
   <set-property
     property="user"  value="root"/>
</data-sources>
```

Figure 4

The proposal of automatically setting configuration of MySQL

Figure 4 describes the automatic load settings, which is set on *false*. By the value setting on *true* the connection by application's installation is automatically created. The next is followed by the description of the database connector's name, of minimal and maximal number of accounts, of login, of password and of address, on which the database is running. The individual configuration values are described in documentation of design pattern. If after all of these steps occur

connection with database, model can start the communication with database with the help of query commands, which are in this case formed by simple order, or by sequence of needful variables definition, which are grouped into one whole with a command *PreparedStatement.* Because this method is implemented in configured file, it increases the possibility of settings asynchronization, and in order to solve these problems on Department of Computers and Informatics, Technical university of Košice the first method of database connection creation is preferred (Fig. 3).

Conclusions

The demandingness of the requirements for realization of the application that are based on standard Model-View-Controller is also nowadays big enough. However, the security of these applications is multiple bigger in comparison with contemporary alternatives for creating of dynamically applications. In the solution of this task was proposed the triple combination of security elements for created application. The merger of these elements forms suitable preconditions for usage of this application in internet. In spite of that MVC is adapted for safe communication in internet, in practise it is mostly used as an intranet application for various organizations. The good adjust of system and suitable combination of secure elements is standard MVC and its extensions are future of dynamically generated web applications.

References

[1] Cavaness, Ch.: Programming Jakarta Struts. O'Reilly&Associates, Inc., Sebastopol, CA, 2003. 468 strán. ISBN: 0-596-00328-5

[2] Gamma, E. - Helm, R. - Johnson, R. - Vlissides, J.: Design patterns: elements of reusable object-oriented software. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1995, 388 strán, ISBN 80-247-0302-5

[3] Hall, M.: Java servlety a stránky JSP. Neocortex splo. S.r.o., Praha, 2001. Dostupné na internete: http://pdf.coreservlets.com (marec 2005)

[4] Vokorokos, L., Jelšina, M.: Počítače základy technických prostriedkov, Mercury s.r.o., Košice 2004, ISBN 80-89061-90-7

[5] Herout, P.: Učebnice jazyka Java, Kopp, České Budějovice, 2000, 352 strán, ISBN 80-7232-115-3

[6] Monson-Haefel, R.: Enterprise JavaBeans. O'Reilly&Associates, Inc., Sebastopol, CA., 2001, ISBN: 0-596-00530-X, 788 strán

[7] Marinescu, F.: EJB Design Patterns, WILEY & SONS, Berlin, 2002, 259 strán, ISBN 0471208310

[8] Java technology documentacion, http://www.java.sun.com/, apríl 2005

Supported by VEGA project No. 1/1064/04