

Model-Controller Interfacing using Struts-Based Web Application

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Computer Science

By

Deepti Bhardwaj

Spring 2011

©2011

Deepti Bhardwaj

ALL RIGHTS RESERVED

SAN JOSÉ STATE UNIVERSITY

The Undersigned Thesis Committee Approves the Thesis Titled

Model-Controller Interfacing using Struts-Based
Web Application

By
Deepti Bhardwaj

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

Dr. Chris Pollett, Department of Computer Science 05/17/2011

Dr. Robert Chun, Department of Computer Science 05/17/2011

Dr. Agustin Araya, Department of Computer Science 05/17/2011

ABSTRACT

“Model-Controller Interfacing using Struts-Based Web Application” is an IDE, named as StrutsHib IDE, for interfacing the model and controller of a MVC architecture based applications. It is developed using Java based technologies such as Struts framework, Hibernate and jQuery. The StrutsHib IDE maximizes the programmer productivity and is available across the network. It offers features which makes the creation of the model and controller component of MVC effortless and time effective. Moreover, it automates the process of database creation for a web application which means the user does not have to be a database expert when dealing with databases. Finally, StrutsHib IDE takes all the advantages of a web based application and provides lot of flexibility to the users.

ACKNOWLEDGEMENTS

I would like to take the opportunity to thank my advisor, Dr. Chris Pollett, for providing his constant guidance and support throughout this project. I appreciate my committee members, Dr. Robert Chun and Dr. Agustin Araya, for their time and suggestions.

I would also like to thank my classmates Tejasvi, Swathi and Sugi for their support and feedback during my project.

Table of Contents

1. Introduction.....	10
2. Technologies Used.....	13
2.1. Struts 2.0 Framework.....	13
2.2. Hibernate Framework	13
2.3. jQuery Framework	14
2.4. CKEditor	14
2.5. Firebug Debugger	14
3. Preliminary Work.....	16
3.1. File Creation Application.....	16
3.2. Study design architecture of the web-based IDE “Aurorasdk”	18
3.3. Application to create Model.....	20
3.4. Interface Model and Controller components.	23
4. Project Design.....	25
4.1. Features	25
4.2. Struts Directory Structure	27
5. Implementation	31
5.1. Web Development Challenges.....	31
5.2. Application Flow	32
5.3. Database.....	32
5.4. User Authentication Process	34
5.5. Project Creation	34
5.6. Model Creation	36

5.7. Edit Model Schema.....	36
5.8. Models Association.....	37
5.9. Editing Modes.....	38
5.10. Main Page and Other Features.....	39
6. StrutsHib IDE Testing Phase	42
6.1. Performance	42
6.2. Usability Testing.....	44
7. Conclusion	46
8. References.....	47

Table of Figures

Figure 1. Input form for user's input	16
Figure 2. On submit without file name input, error message is displayed	17
Figure 3. Aurorasdk IDE home page.....	20
Figure 4. jQuery draggable method.....	22
Figure 5. Drag and drop a field to connect two tables.	22
Figure 6. "has a" relationship is denoted by a line with diamond on the tip.....	22
Figure 7. "Is a" relationship, Table 2 is a sub-class of Table 1.....	23
Figure 8. On right click; a menu is displayed with the list of actions.....	23
Figure 9. Struts Web Application Directory Structure.....	28
Figure 10. struts-config.xml – Action Mapping.....	29
Figure 11. web.xml - Servlet Mapping	29
Figure 12. Welcome Page of StrutsHib IDE.....	34
Figure 13. Create New Project Pop-up window.....	35
Figure 14. Create New Model Interface	36
Figure 15. Editing Mode Panels	37
Figure 16. Model Association Interface	38
Figure 17. Association's type drop-down	38
Figure 18. Model File Opened in Edit Mode Panel	39
Figure 19. Main Page of StrutsHib IDE.....	41

Figure 20. Performance Chart – Eclipse vs. StrutsHib vs. Aurorasdk43

Figure 21. Usability Test Results45

1. Introduction

There are number of Integrated Development Environments (IDE) available on the market, for example: JDeveloper, JCreator, NetBeans and Eclipse which provide the environment to develop Java based applications. However, these IDEs are desktop-based applications. Moreover, users have to make sure that these IDEs are properly installed and configured in their machine before start using them, which takes significant amount of effort and time. Even using a properly installed IDE takes time to load. In addition, the user has to take the hassle of managing his project workspace himself; and he can lose the work if his workspace accidentally gets corrupted.

A web-based Integrated Development Environment (IDE) provides the user the ability to build an application quickly without doing much of hand-coding. The client only needs a web browser and an internet connection. The client browser downloads the IDE from the server which enables the client to access prior code saved in the database or create new projects. A web-based IDE overcomes the inconvenience involved in installing and configuring the standard desktop-based IDEs. The most exciting benefit of using a web-based IDE is that, with so many new PDAs and smart phones which support Java and Ajax technologies on the market, programming tasks can be performed anywhere, anytime, and even while traveling.

Some of the web-based IDEs currently available are Mozilla Skywriter or Bepin [10], and Aurorasdk IDE [3]. Bepin is developed by Mozilla Labs and provides a web-based interface for developing applications in HTML, JavaScript and CSS. It is still not in

a mature phase and it has some limitations; such as it only has support for the Mozilla Firefox browser. On the other hand, Aurorasdk is a web-based IDE for Java application inspired by the desktop-based Eclipse IDE. The architecture and features of Aurorasdk IDE are discussed in detail later in this report under Section 2.2. Including Bepin and Aurorasdk, all the other web-based IDE provide just an editor for the application source code with features to help in coding. However, users still have to follow the hand coding approach to develop an application. In this project, the web-based IDE facilitates the generation of source code for the application which saves time consumed by hand coding.

“Model-Controller Interfacing using Struts-Based Web Application” web-based IDE is a Struts application named as StrutsHib IDE. Struts provides a framework for Java Servlets and JSP applications that use Model-View-Controller (MVC) architecture. Specifically, where the Model is a business logic or business process such as Java Beans, the View is dialogs such as Java Server Pages and the Controller is a central control unit such as Java Servlets [6]. This project automates the process of developing the Struts application components. It provides a platform for the rapid development of Model-View-Controller (MVC) architecture based applications.

For this project, I was responsible for implementing the model and controller components of the web-based IDE. Tejasvi Palvai, another Computer Science graduate student, implemented the view component [15].

StrutsHib IDE provides an easy and convenient way to create a model component for MVC architecture based applications. It also helps the user to interface and integrate the model and controller components of a web application. Using StrutsHib IDE, developers

can map the client or browser requests to an appropriate action, which implements the model in the controller. All these operations can be done by just doing some clicks and drag-drop functionality provided by this web-based IDE.

Using StrutsHib IDE, the user does not need to have a good knowledge of SQL queries for the creation of relational databases. The user can create database tables with the interface provided and StrutsHib IDE can perform the necessary queries in the background to create the exact structure on the database server. It also provides a functionality to drag the model object and drop onto one of the controller methods for interfacing. This StrutsHib web-based IDE saves the time involved in hand writing the code for connecting the controller and model components.

In this report, the technologies involved in this project are discussed in Section 2. Mostly, all the technologies used are very well accepted by the developer community. The preliminary work done in order to start this project is discussed in Section 3. Section 4 of this report, kicks off the discussions about the project design and features available. Further in detail, Section 5 covers the implementation details along with the challenges faced while working on this project. As we evaluated StrutsHib IDE with some other available web-based IDEs, Section 6 shows the results of the performance and usability testing done on it. Lastly, the conclusion of the project is presented in section four.

2. Technologies Used

StrutsHib IDE is created using advanced Java technologies such as Struts 2.0 [6] and Hibernate framework [11]. For the client side interactivity features, we have used jQuery, a JavaScript framework [7]. While working on StrutsHib IDE, we came across the requirement to provide a text editor to the user for editing the source code. Fortunately, an open source project CKEditor that fulfills this requirement and integrated well with our IDE [13].

2.1. Struts 2.0 Framework

Struts provides a framework for Servlets and Java Server Pages (JSP) applications that use the Model-View-Controller (MVC) architecture where the Model is a business logic or business process, such as Java Beans, View is dialogs, such as JSP and HTML pages, and the Controller is a central control unit, such as Java Servlets [6]. We have used the Struts framework to implement our project. StrutsHib IDE provides the functionality to develop the MVC architecture based applications.

2.2. Hibernate Framework

Hibernate is an open-source framework written in Java by the Red Hat organization [11]. It has libraries for object-relational mappings. These libraries are java classes which facilitate the interaction with the databases. Using Hibernate, the programmer can focus on the logic without worrying about the database connections and result-set handling.

The StrutsHib IDE automatically generates the Model methods using Hibernate java classes and its methods. We decided to use Hibernate to give the user the flexibility to

select the database at any point of time while developing his project. The change in desired database on the backend only requires the configuration changes without any change in the source code or database queries.

2.3. jQuery Framework

jQuery provides the capabilities to create plug-ins on top of the JavaScript library [7]. jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. It contributes to the creation of powerful and dynamic web pages. We used jQuery UI 1.8.5 libraries in our project to provide drag and drop features to facilitate the development process of an application. jQuery plug-in context menu is used to implement the right-click menu options for the GUI components. These context menu options open up on right - clicking on the elements- with the options related to different operations.

2.4. CKEditor

Our StrutsHib IDE provides the edit mode in which the user can manually edit the source code of the files. These source code files can be a model, view or a controller file of a project created by a user. We have used CKEditor version 3.4, an open source rich-text editor, which allows users to easily edit their source code [13]. StrutsHib IDE has used it as a plug-in to save a file and to provide an auto-suggest feature.

2.5. Firebug Debugger

The Firebug debugger add-on for Mozilla is used to debug the HTML, CSS and JavaScript source code during the implementation. It is a powerful and free add-on feature

for the Mozilla Firefox web-browser. The Firebug tool helps the developer to inspect the individual element of the DOM tree, along with the CSS attached to it. In our project, it is used to debug the code which helped in the rapid development of the project.

3. Preliminary Work

Before starting to working on the StrutsHib IDE implementation, we studied and evaluated the Java technologies available for the web development. In this section of the report, we discuss about the different web applications developed in order to understand the frameworks used in the implementation of our StrutsHib IDE. In addition to learning the new technologies, we studied the architecture of the existing web-based IDEs. Section 3.2 discusses the architecture of the web-based IDE “Aurarasdk”. These preliminary exercises proved to be a great help while working on this project.

3.1. File Creation Application

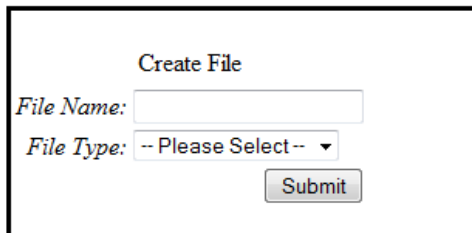
An application for creating files on the server and storing an entry into the database table is developed. The goal of this application is to implement a simple Struts-based application. Struts2.0 framework, which follows the MVC architecture, is used to design the architecture of the application. MySQL database is used on the backend to store the file details. It helped to understand the Struts framework.

In the View component of the application, Struts 2.0 tags are used to create the form elements [8]. On submit, an Ajax call is made for an action class which interacts with the model component to create the file. The action class returns ERROR in case of any validation error otherwise SUCCESS. This application provides the user an interface which takes file name and file type as inputs. On submit, an application validates the inputs and in the case of failure, displays the error on the same page. In the case of the valid inputs, it

creates the blank file with the name and type provided on the server. The application also inserts the information of the file created into the database table for future reference.

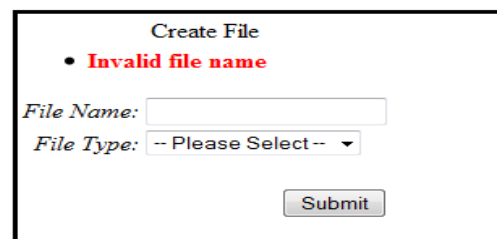
On request, the server looks up the action mapping in struts.xml and invokes the mapped servlet. The input form page is displayed on the client browser. On submit, an Ajax call is sent and based on the mapping in struts.xml, execute () method of the action class is called. Based on what the execute method returns, the framework determines which page needs to be called from the mapping in struts.xml.

In the case of an empty string for a file name, “*Invalid file name*” error message is displayed on the screen.



The screenshot shows a web form titled "Create File". It contains two input fields: "File Name:" followed by a text input box, and "File Type:" followed by a dropdown menu with the text "-- Please Select --". Below these fields is a "Submit" button.

Figure 1. Input form for user’s input



The screenshot shows the same "Create File" form as in Figure 1, but with an error message displayed at the top: "• Invalid file name". The "File Name" input box is empty, and the "File Type" dropdown menu is still set to "-- Please Select --". The "Submit" button is visible at the bottom.

Figure 2. On submit without file name input, error message is displayed on the same page.

For not selecting file type from the drop down, the “*Invalid file type*” error message is shown to the user. When both the input file name and file type are valid inputs, a file is created on the server and a success message, for example, “*File Created Test.jsp*” is displayed.

Figure 3 shows a web form titled "Create File". At the top left, there is a red bullet point followed by the text "Invalid file type". Below this, the "File Name" field contains the text "TestFile". The "File Type" dropdown menu is currently set to "-- Please Select --". A "Submit" button is located at the bottom right of the form.

Figure 3. On submit without selecting file's type, error message is displayed on the same page.

Figure 4 shows the same "Create File" form. The "File Name" field contains "TestFile" and the "File Type" dropdown menu is now set to "JSP". The "Submit" button is visible at the bottom right.

Figure 4. For valid inputs, a file is created on the server.

The error handling techniques from this section are used during the implementation of our StrutsHib IDE. Thus, this exercise proved to be useful for our project design and implementation.

3.2. Study design architecture of the web-based IDE “Aurorasdk”

The goal is to study the architecture of the web-based IDE “Aurarasdk” [3]. Aurora SDK is an IDE for developing java based applications. The reason behind the study is to understand the architecture of this web-based IDE which helped in designing the architecture of our StrutsHib IDE. Also, the study provided the limitations of Aurara SDK limitations which we improved in our IDE.

Aurorasdk web application is mainly implemented using Google Web Toolkit (GWT), Servlet and MySQL. This project is done under the guidance of Professor Gail C. Murphy at University of British Columbia. Aurora SDK supports full syntax highlighting and full compilation of Java source codes with different plug-in compilers. Two compilers are implemented in Aurora SDK: Eclipse and JDK. It also supports auto-completion and

provides an environment to run the compiled code. The architecture used for the IDE has the following four layers which makes the design flexible:

- Application layer
- Service layer
- Compiler layer
- Persistence layer

In the Application layer, GWT widgets are built to develop UI. In the Service layer, GWT Remote Service is used to communicate between different layers. Two compilers, Eclipse and JDK, are implemented in the Compiler layer. Aurora SDK uses MySQL database as a data store- which is in the Persistence layer. There are some issues with Aurora SDK web-based IDE which need to be addressed. It does not support concurrent compilation. The idea compilation feature breaks and generates incorrect compiled code when two users compile the files at the same time. The editor of the Aurora SDK does not work properly in Internet Explorer browser.

In future releases, Aurora SDK team plans to enhance the IDE with new features. They want to allow multiple users to work on a single project to encourage collaboration at work place. The team is planning to provide a functionality to debug the source code. In addition to Eclipse and JDK compilers, they want to plug-in more compilers.

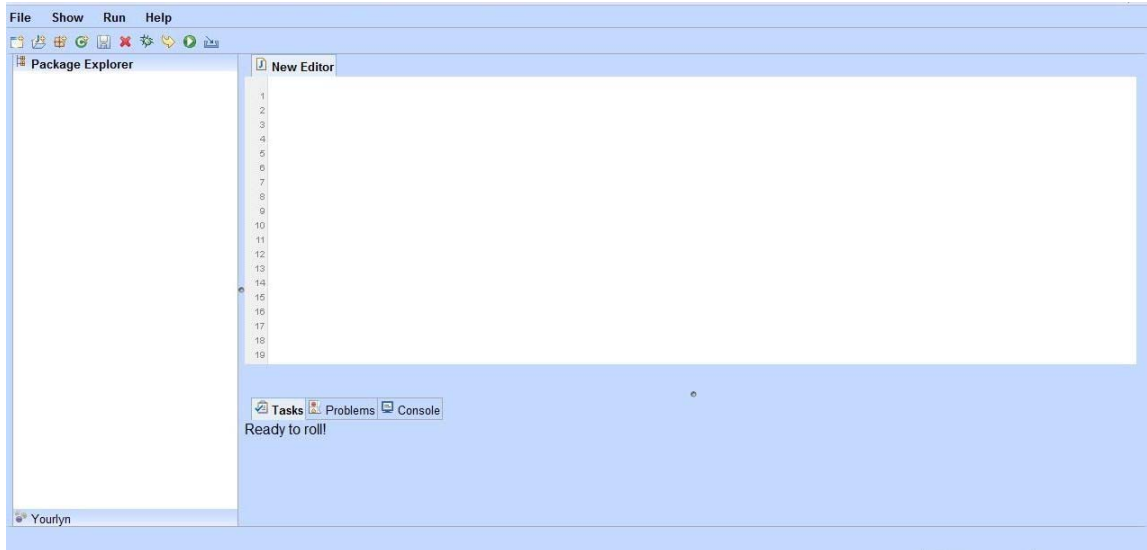


Figure 3. Aurorasdk IDE home page.

3.3. Application to create Model

An application for creating model components is developed using HTML elements and JQuery framework. It also provides functionality to perform different operations on the model object. The animation libraries are used to build animations such as: drag and drop; drawing lines between tables; and a right-click pop-up menu. The application allows the user to add fields to the table. It has a feature for creating relationships between the tables to build a model component. The user can delete the table and rename it. The goal is to learn jQuery and develop an application using it. This application helped to visualize the implementation of the IDE component to create models. It has a menu bar to create a new table. It also has a right-click on table object and a pop-up a menu list of various operations that can be performed on the object. The application provides an option to drag the table objects within the main frame. In this application, the following two types of relationships are supported:

- Composition: “has a” relationship.
- Sub-class: “Is a” relationship.

The application uses JQuery framework for drag and drop animations. It includes the “*wz_jsgraphics.js*” JavaScript library which is available online for drawing lines and different type of shapes. On request, the browser renders the HTML code and displays the index page. The web page has the menu on the top allows the user to create new table. When the user clicks the “New Table” link, a new table is created with default name. A menu list is displayed on right-click on the table created. The user can add new fields into the table by clicking the “Add Field” option from the pop-up menu. The “Delete” option is used to delete a particular table along with its fields. To rename the default table name, click on “Edit” and a textbox is displayed. After entering a new name, press enter and the table renames to the value entered. This application takes advantage of jQuery features by providing the dragging of the table object. The table object created has a `draggable ()` method which is called when the user drags the table. It also has a `droppable ()` method which is called when a field is dropped on it. The fields added to the table are draggable fields. They can be dragged and dropped onto another table and fields of another table. When a field is dragged a clone of that field, it is generated and dragged, instead of the original object. The below is a code snippet of `draggable` and `droppable` methods of a field object.

```

$('.attdiv').draggable ({helper:'clone'}).droppable ({
    drop: function (ui, event) {
        var source = $(event.draggable).clone ().attr ("id");
        var dest = $(this).attr ("id");
        drawLineBetweenBox (source, dest);
    }
});

```

Figure 4. jQuery draggable method.

Below are screenshots of the functionalities implemented to learn jQuery framework in detail. These experiments to develop different applications provided a great level of confidence with which to use the technology in our project.

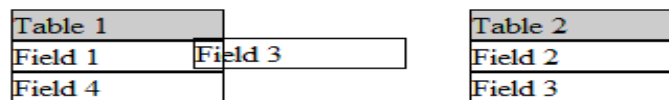


Figure 5. Drag and drop a clone of field from one table to another to connect two tables.

In database design and object oriented program architecture, “has-a” is a relationship where one object is a part another object.

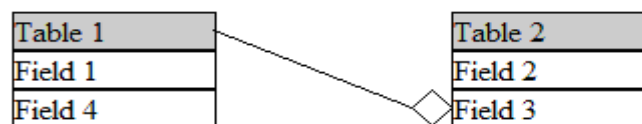


Figure 6. Composition or “has a” relationship is denoted by a line with diamond on the tip.

“Is a” is a relationship where one class is a subclass of another class. These relationships are used to connect different tables in order to create a Model object.

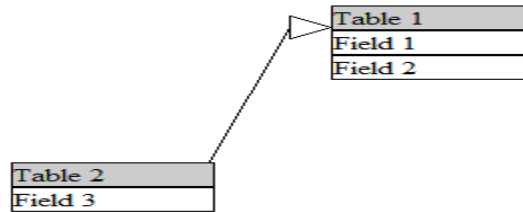


Figure 7. “Is a” relationship, Table 2 is a sub-class of Table 1.

In addition to above, the application provides following functionalities to the user:

- Add new fields to the tables.
- Edit the default name of a table.
- Delete a table.

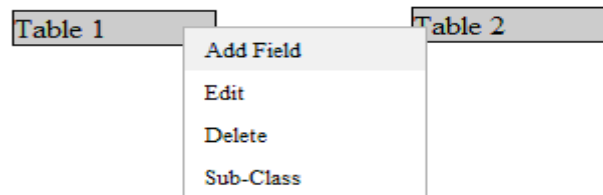


Figure 8. On right click; a menu is displayed with the list of actions.

The similar functionality of drag and drag, we implemented in our StrutsHib IDE.

The application done in this section saved time as we reused the code written for drag and drop; and right-clicks menu for StrutsHib IDE.

3.4. Interface Model and Controller components.

The objective is to build an application functionality to interface the Model and Controller components. This interfacing is highly important for the MVC based applications. The Controller list is available to the user along with the all method names. The user has to create a Model object and decide to which Controller method it should be connected. Just dragging and dropping the Model object connects the Model object with

the Controller. The list of Controllers on the web page can be expanded or collapsed. The menu list to display the Controllers is implemented using CSS styles. JQuery library methods are used for drag and drop functionality. An available online JavaScript library is used to implement the expand and collapse functionality. On page load, the list of Controllers is displayed to the user on the right hand panel. Initially, the list is in a collapsed state which means it just displays the names of the Controllers. When the user clicks on the expand icon, `expandTree ()` method is called which takes the element id as input. On clicking the collapse icon, it calls `collapseTree ()` method to collapse the list.

4. Project Design

The objective of our project is to provide users a web-based tool which is easy to use and help them in the rapid development of applications. Users should get the flexibility to use it anywhere anytime. They should not worry about the source code files and database setup for their applications. Also, our StrutsHib IDE should provide the critical features required for developing MVC architecture based applications.

4.1. Features

The IDE provides the below mentioned features to the users for developing an application based on MVC pattern and backend driven. Some of the features StrutsHib IDE provides are: user registration; project creation; model and controller file creation; interface to edit the model schema; and associating the model and controller files. These features are related to Model and Controller interfacing, along with some common functionality and discussed in more detail below.

Login Page and User Registration

The login page is provided to authenticate the registered users. It is an entry point to the IDE where the user inputs the username and password. After authentication, the user gets the access to his previously saved projects. The user can either work on his existing projects or can create new projects to work on.

Create Project

A simplified interface is provided to the user to create projects. The IDE automatically generates the default structure for the project. The user does not have to

concern about the default settings required between the application components and the backend database.

Create Model

The Model component of the MVC pattern is a component which interacts with the backend application or resource. On the backend, there can be any relational database. Using this web-based IDE, the user does not have to learn Structured Query Language (SQL) in order to create tables and perform different operations. Using the widget IDE provides the creation of the Model and a default schema is created inside the selected database for the user automatically. In this IDE, hibernate technology is used to connect to the database. When creating a Model, the default hibernate methods are saved in the Model file.

Edit Model schema

Our IDE provides the interface to edit the default schema generated at the time of Model creation. The user does not have to learn the SQL alter command to modify the schema of the table. The IDE eliminates the need of an administrative tool required to manage and execute any SQL query for the database tables. It not only generates the alter SQL query to modify the schema but it also executes the query to actually do the alteration.

Models Association

One of the critical features of the IDE is to link different model components of an application. There is an IDE interface available for the user to generate the linkage between models. The linking is called as an association between the models. The IDE presently supports one-to-one, many-to-one and one-to-many associations. These associations are

discussed in section 3.3. Using this feature, the user gets the advantage of time and effort. The user does not have to create these relationships or associations between different database tables manually through SQL queries.

Create Controller

The user can add a new Controller to his project using the interface available on the IDE. When a new Controller is added to the project, the IDE automatically generates the basic structure for the Controller file and link the file to the project. The file is automatically saved inside the Controller directory of the project.

Interfacing Model and Controller

In MVC, the Controller component interacts with the Model class whenever the View triggers the request. The IDE provides an easy mechanism to implement the interfacing between the model and the controller. The user can drag and drop the Model file onto the Controller file to do the association. In addition, the IDE provides the auto suggestions of the Model methods when the user edits the source code of the Controller file.

4.2. Struts Directory Structure

The below Figure 11 demonstrates the standard directory structure of an application based on Struts framework. Struts framework has three main components: Model, View and Controller. The files associated with different components are arranged in different directories under the project folder.

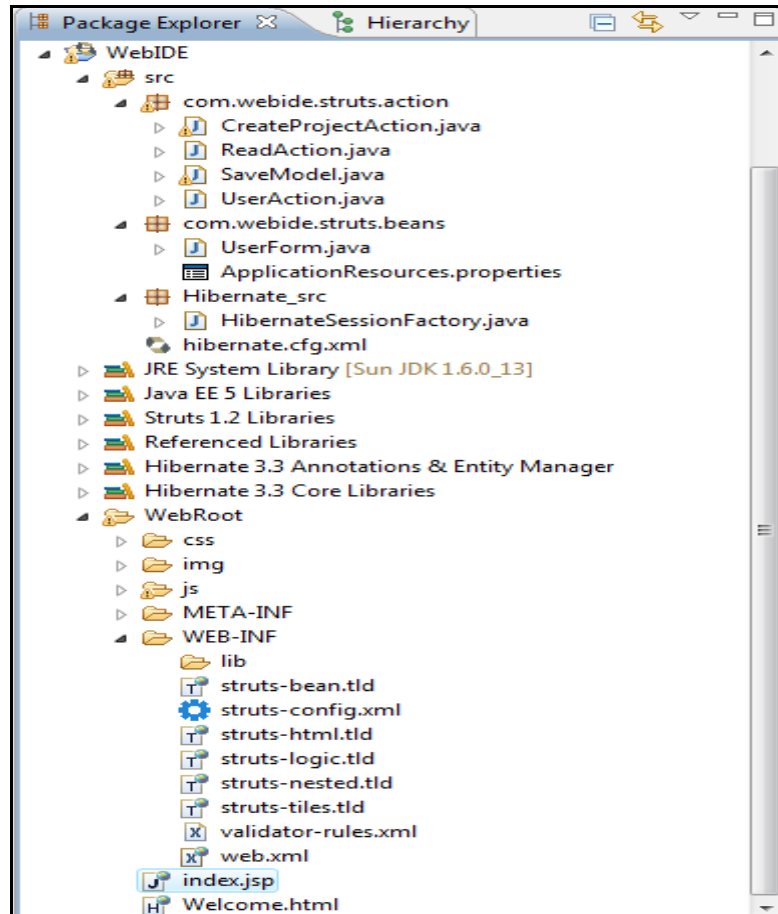


Figure 9. Struts Web Application Directory Structure

Action Class Files

The controller component of Struts receives the request from client browser in the form of *Actions* and these actions are defined in the struts-config.xml file. For this particular project, the action class files are placed under the well-defined package named *com.webide.struts.action*.

Action Form Files

When the user submits the input form, the view component passes the input values to the model component using the *Action Form* java beans. The action forms are packaged under the *com.webide.struts.beans* for this project.

Struts-Config.xml File

The configurations file *struts-config.xml* is used to define all the binding information for the different components of the Struts framework. It routes the request coming from a client or web-browser to the defined handler. This *struts-config.xml* is placed under the *Web-INF* directory along with other configuration files. The following Figure 12 shows one of the action-mappings done inside the *struts-config.xml* file:

```
<action-mappings >

  <action name="UserForm" path="/User"
    type="com.webide.struts.action.UserAction" validate="true"
    input="Welcome.html">
    <forward name="success" path="/index.jsp"/>
  </action>

</action-mappings>
```

Figure 10. *struts-config.xml* – Action Mapping

Another critical configuration file is *web.xml* which stores the information about the servlet. Figure 13 illustrate the most important servlet-mapping which specifies the action which needs to be done when the request handler notices a call from a *.do files.

```
<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

Figure 11. *web.xml* - Servlet Mapping

View Resources

The well structured property of the Struts framework helps the web container to search for the resources effortlessly while building and executing an application. As the

entire set of client side resources are placed under the *Web Root* directory of the application folder. JSP and HTML pages, CSS Stylesheet and JavaScript files, Resource bundles, and tag libraries are the view resources kept under Web Root directory. For this project the most important JavaScript libraries used are: jquery.min.js; jquery-ui.js; and CKEditor files.

5. Implementation

In this section, we have discussed the implementation details of our StrutsHib IDE. As StrutsHib IDE is a web-based Struts application, this section also shares some of the important differences between web development and stand-alone application development.

5.1. Web Development Challenges

Web development involves many more challenges than application development. As web-based applications are always deployed and hosted on a web server or application server, and accessed over network. So, the performance of the web-based application depends on the network speed and internet connection availability. The performance is not a big issue with the desktop-based applications as it's running on the local machine and all the resources are readily available. StrutsHib IDE is a web-based IDE which makes the development process critical. Users are not going to accept a slow and poor performance IDE, so the implementation challenge in the case of StrutsHib IDE was to overcome the performance issues and provide a better user experience. To provide a better user experience, we have used jQuery client side scripting to implement features like drag and drop, and other pop-up windows features.

Another challenge was to optimize the page load time after receiving the response from the server. Thus, instead of a synchronous call to the server, AJAX asynchronous channel is used to communicate with the server. Since the asynchronous call run in the background, the user is free to do any activity on the client side while the response comes back. Only the desired portion of the page is refreshed using the AJAX response. The

validations of the input field values in forms are performed on the client side to save multiple hits on the server.

For desktop-based applications, cross platform compatibility is important; where as for web-based applications, cross browser compatibility is a critical feature. During the development process of StrutsHib web-based IDE, appropriate coding standards are used to make it work on all the new generation popular web browsers. This StrutsHib IDE is developed for and tested against many web browsers such as Mozilla Firefox, Internet Explorer 7 and above, and Google Chrome.

5.2. Application Flow

Java Server Page (JSP) technology is used to implement the view component. On every user action, AJAX technology is used to make the call to request the handler. The controller component processes the request and the response from the business layer is then returned to the view component to display it on the client. The AJAX response is either a string or a JSON object. The StrutsHib web-based IDE can be deployed on any Java based web or application server using a Java war format. Once successfully deployed, it is accessed by invoking the url *http://<host-name>:<host-port>/WebIDE/* in a web browser.

5.3. Database

MySQL database is used as the backend data store for StrutsHib web-based IDE. The database named *web_ide* is used for storing the data for IDE. The following are the database tables used:

Users (*id, Name, Password, UserName*)

The database table *Users* stores the information about the registered users. During the login process, the user is authenticated against the username and password stored in the table. The passwords are stored as hash values in the database for the security purpose.

Projects (*id, projectname, projectpath, user_id*)

The *projects* database table has all the information about the projects created by the users using this IDE. The field named *user_id* is a foreign key to *Users* table which links the project to the owner of the project. *Projects* table has *projectpath* which stores the physical path of the project directory on the server.

Models (*id, modelname, modelpath, project_id*)

The *models* table stores the information about the model components that the user created of a particular project. The foreign key *project_id* links the model to a project in the *projects* table. The table field *modelpath* stores the package path of the model class which is required to locate the file.

Models_Association (*id, model_id1, model_id2, asso_type*)

The database table named *Models_association* stores the user created associations between *model_id1* and *model_id2*. This table has a foreign key relationship with the *model* table. The *asso_type* field stores one of the three status types: 'belongsTo', 'hasMany' and 'hasOne'.

Controllers (*id, controllername, controllerpath, project_id*)

The *controllers* database table stores the information about all controller files for every project. The field *controllername* is the name of the controller file and *controllerpath*

is the path on the server where the controller source code file is stored. The foreign key *project_id* points to the project record in the *projects* table.

5.4. User Authentication Process

When the user accesses the application root URL: *http:// <host-name>:<host-port>/WebIDE/*, the request is redirected to the *Welcome.jsp* page. The *web.xml* file contains the redirection logic. Figure 14 shows the login page which is displayed to the user. It is an entry gateway to access the IDE features. The user visiting for the first time is provided with the option to register and follows the “New User?” link. The StrutsHib IDE also provides a password recovery feature to the user. The user is authenticated using the username and password provided. When the user clicks the submit button, the request is send to the controller to handle. The user is authenticated against the entry saved in the *users* database table.



San José State University

Web Based IDE

Please enter your username and password

Username

Password

[New User?](#) [Forgot Password?](#)

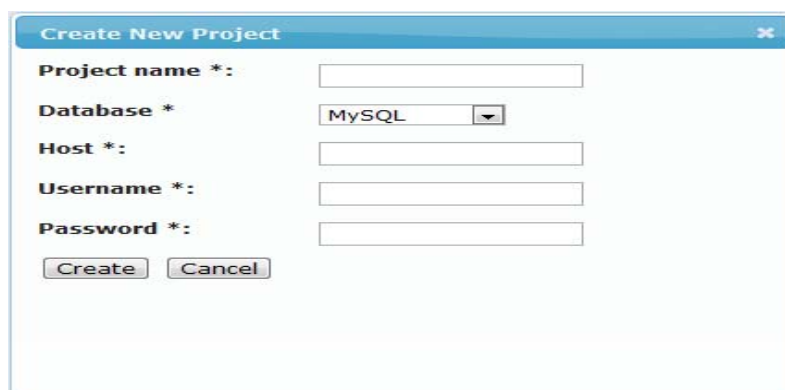
Figure 12. Welcome Page of StrutsHib IDE

5.5. Project Creation

Once the user is authenticated, the main page of the application is displayed. There are two ways to create a new project. One way is to click on the item “New Project” from

the “File” menu option provided on the top menu bar. Another option is to click on the “Create New Project” button provided on the left hand panel. Both actions pop-up the form shown in Figure 15 for the user inputs. When the user performs the project creation action, it is handled by the *CreateProjectAction.java*. This action class extends the functionality of the Struts action class *org.apache.struts.action.Action*. The model object is then invoked to create the project on the server and makes an entry in the *projects* database table. jQuery methods are used on the client-side to handle the on-click event to pop-up the create project form widget. On clicking “Create” button, an Ajax call is sent using the jQuery Ajax method. Ajax call is handled by a controller class to invoke the appropriate model objects.

During this process, many operations are performed automatically by the application. The application creates a project directory along with default MVC templates with the name provided by the user. A database is created for the user automatically based on his inputs. The cancel button and the cross [X] icon on the top right are provided to the user to abort the project creation process. This operation hides the creation widget and displays the main page.



The image shows a web browser window titled "Create New Project" with a close button in the top right corner. The window contains a form with the following fields and controls:

- Project name *:** A text input field.
- Database *:** A dropdown menu with "MySQL" selected.
- Host *:** A text input field.
- Username *:** A text input field.
- Password *:** A text input field.
- Buttons:** "Create" and "Cancel" buttons at the bottom left.

Figure 13. Create New Project Pop-up window

5.6. Model Creation

When a user creates a project, StrutsHib IDE creates a default project template. A default model hibernate class is created for the user with the same name as the project name. However, our StrutsHib IDE also provides a feature to add new model classes to the project. The user gets a right-click menu option “Create New Model” on the *Models* folder of every project. The interface to create the model class is simplified with just one input field. The model name value that the user enters is passed as an input to the *SaveModel.java* action class. The action class executes the logic to create a hibernate class with the default template code in it. The generated hibernate class is created under */Projects/<project folder name>/Model/* directory structure.

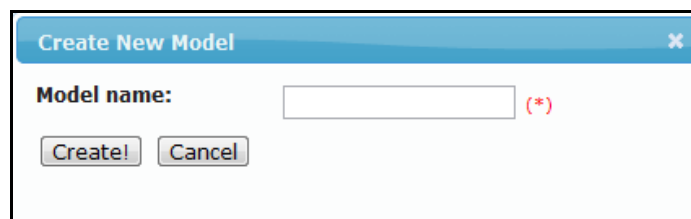


Figure 14. Create New Model Interface

5.7. Edit Model Schema

The model schema can be modified at any time by the user. StrutsHib IDE gives users the capability to do the schema changes without writing any SQL queries. It provides a widget on the right-click option for the model class to edit the schema. The following operations can be performed:

- Add new fields to the model database table.
- Delete the existing fields from the model database table.

- Edit the name, data type or size properties of a field.

After performing any of the above operations, when the user clicks the “Save Model Schema” button, the changes are made to the database tables. Also, the Hibernate class methods for that model are updated according to the new values. The default hibernate class methods *public Integer getID ()* and *public void setID (Integer ID)* can also be removed or replaced with any user defined methods. These methods are representing the *id* field of the model object. Figure 17 shows the edit model schema interface which shows up on clicking the edit model schema option. The delete operation is a little tricky as the application has to check for the dependencies from the other database tables.

Design Mode		Edit Mode	
id	integer	10	[X]
name	string	100	[X]
address	string	255	[X] Delete Field
phonenumber	integer	10	[X]

Figure 15. Editing Mode Panels

5.8. Models Association

StrutsHib IDE has an interface for associating the models of a particular project. It supports three types of relationships or associations as shown in Figure 19. A right-click on

the model file presents an option “Associate w/ Another Model” and on selecting this option the interface in Figure 18 is displayed to the user. The user can select relationship type from the dropdown and the model name from the other drop down. On clicking “Create new association” button, an Ajax call is made for the *CreateModelAssoAction.java* action class. The action class then invokes the appropriate model object to make the entry in the database and create the hibernate methods.

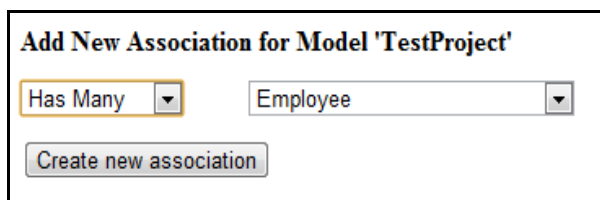


Figure 16. Model Association Interface drop-down

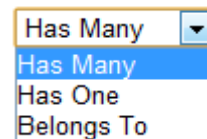


Figure 17. Association's type

5.9. Editing Modes

There are two modes or interfaces that IDE provides to edit the source code of a file: Design Mode and Edit Mode. Figure 17, below, illustrates the design mode for the model class and Figure 20 shows the edit mode for the same model class with a default template.

For the design mode, the jQuery methods are used to provide features like drag and drop; and they add new DOM elements. The below jQuery source code lines below demonstrate the drag and drop features for model and controller components.

```
$("#div.model").draggable ({helper: 'clone', start: function (event, ui) {}});
$("#div.controller").droppable ({drop: function (event, ui) {...}})
```

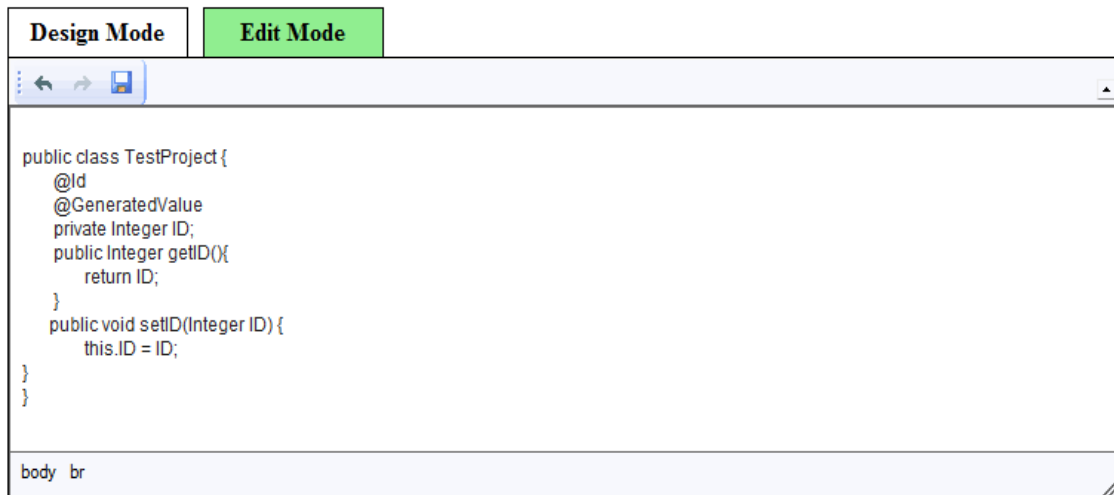


Figure 18. Model File Opened in Edit Mode Panel

For the edit mode, CKEditor is used to display the source code of the files. It allows the user to edit the code and save the edited code. When the user clicks the “Edit Mode” tab, the application hides the design mode panel and switches the focus to the edit mode panel. The below jQuery code lines below hide the design panel and switch to the edit panel, when the edit mode click is captured.

```
$('#tbl_design').hide ();
SW.formsRegulator.switch_form ('editpanel');
```

5.10. Main Page and Other Features

Figure 21 is the main page of our StrutsHib IDE. It is displayed to the user once the authentication is done. The main page captures the user name of the currently logged in user, using session variables, and displays the welcome message on the top of the page. The main page is divided into four important panels which are: top panel, left panel, middle panel and right panel. The top panel of the page has menu options which are *File* and *Help*. The menu option *File* has a submenu with *New Project* and *Logout* submenu items. *New*

Project option is used to create the new projects and *Logout* as name suggests is to logout from the application. When the user clicks *Logout*, the current session of the user is destroyed and it takes the user to the welcome page. The *Help* option provides a detailed user guide about the application features and functionalities.

The left panel is named Projects and it displays the list of projects. The list of projects is associated with the current logged in user. The list of projects is initially collapsed and a click on the project name expands the project folder. The script *treeview.js* is used to implement the collapsible and expandable properties to the `` html list. All the `` elements of the list are linked with the right-click menu options. The project name has an “Export Project” option on right-click. The model folder gets the “Create New Model” menu option on right-click and this feature is discussed in detail in Section 5.5 of the report.

The middle panel has two modes of editing a file and this feature is discussed in Section 5.8. The right panel on the page is a very useful panel for the users. It is named *Help* panel and as the name suggests, it provides the help guide to the user. This help panel is different than the Help option in the top menu. The *Help* in the main menu provides the functional details of all the features that IDE provides; whereas the *Help panel* on the right is associated with the user action. This help panel shows the help text related to the action performed. It is an interactive interface which responds to the user action and provides a better user experience. For example, in the below Figure 21, the user has selected “Edit Model Schema” option and the help text related to the action performed is displayed. Help

panel provides all the necessary information required to use the “Edit Model Schema” design mode interface.

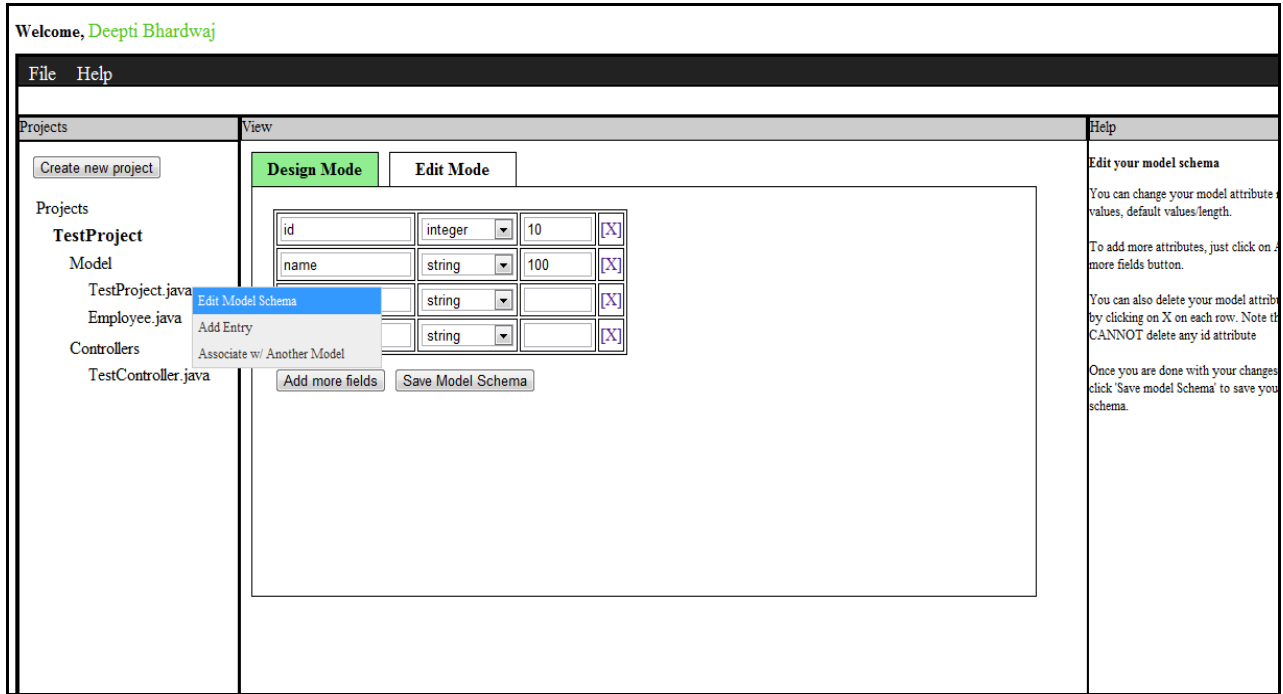


Figure 19. Main Page of StrutsHib IDE

6. StrutsHib IDE Testing Phase

Software testing is an essential phase of a software life cycle. The testing is done to make sure all the intended features are working as expected and to resolve the bugs discovered during the process. The testing is done in several phases such as: unit testing, browsers compatibility test, performance and usability testing. In this section, the testing process and results are discussed.

During unit testing, the application is tested against the functional and non-functional test cases. The functional test cases are performed to compare the behavior of all the implemented features with their expected behavior. The non-functional tests involve the process of testing the StrutsHib IDE on different platforms or environments. StrutsHib IDE is tested on Windows and Mac platforms and it can be deployed on any of these tested platforms without any issues. In our project, we have used the *Firebug* debug tool during the front end development process. Since Firebug is a plug-in for Mozilla Firefox browser, *Firefox* is the web browser used during the development of StrutsHib IDE. However, our StrutsHib IDE is tested on other popular web browsers such as: Internet Explorer (version 6 and above), Google Chrome and Safari. It is compatible with all these browsers; although the styling of the pages gets slightly changed based on the browser used.

6.1. Performance

To test the performance of our StrutsHib IDE, we calculated the time taken to perform certain tasks. The results were evaluated against the performance of existing IDEs such as Eclipse and Aurorasdk IDE. For the performance test, we selected Eclipse and

Aurorasdk because both are Java based IDEs. Where Eclipse is a desktop-based IDE, Aurorasdk is a web-based IDE. We downloaded and installed both Eclipse and Aurorasdk on the same machine where the StrutsHib IDE was deployed. Since all three IDEs were running on the same hardware and environment during the test, the test results are unaffected by these factors. Figure 22 shows the performance testing. The chart below is plotted using the set of activities performed on all the three IDEs and the approximate time taken by the IDEs to complete these activities.

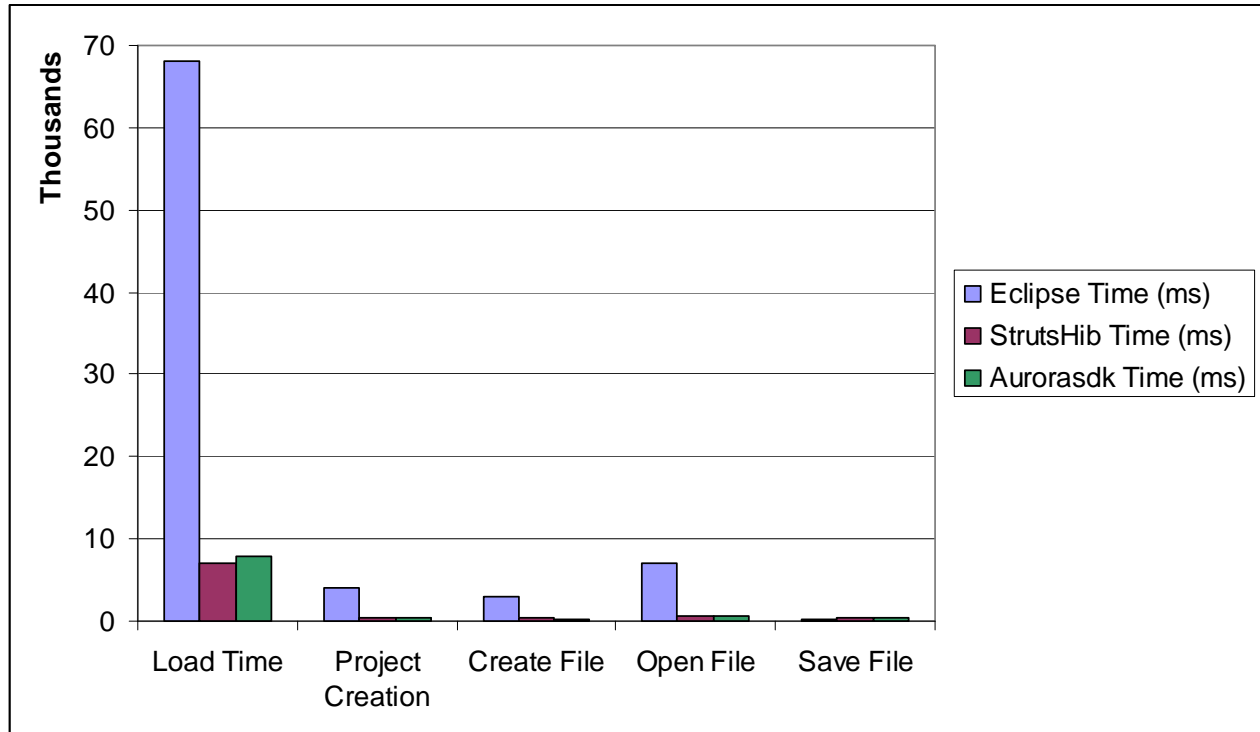


Figure 20. Performance Chart – Eclipse vs. StrutsHib vs. Aurorasdk

In the above chart, the results clearly show that the web-based IDEs take lesser load times than the Eclipse desktop-based IDE. There is a huge difference between the times taken by the Eclipse desktop-based IDE to load than by the other two web-based IDEs.

Other tasks such as project creation, create file and open file, can be performed in less time with web-based IDEs than Eclipse. However, for the save operation, Eclipse, performed better. According to the results, the performance of StrutsHib and Aurorasdk are comparable.

6.2. Usability Testing

Usability testing involves the process of testing with actual users of the system. The usability testing for our StrutsHib IDE is conducted with three expert and three novice users. The purpose of testing with expert and novice users is to analyze how they perform assigned tasks on StrutsHib IDE. The developers that use other IDEs for coding are considered as expert users and users who are new to the IDEs are novice users. The following tasks are assigned to all the users and we analyzed their performance.

Task 1. Create a user account and login using username and password.

Task 2. Create your new project 'TestProject' and browse the files structure created by the StrutsHib IDE.

Task 3. Create a new model 'TestModel' under the Model directory and open the file in the design mode.

Task 4. Edit the model schema by adding new fields to it using the design mode editor.

Task 5. Create an association between the 'TestModel' model component and default model 'TestProject' using design mode editor.

Task 6. Create interfacing between the 'TestModel' model and controller component 'TestController' using the drag and drop feature.

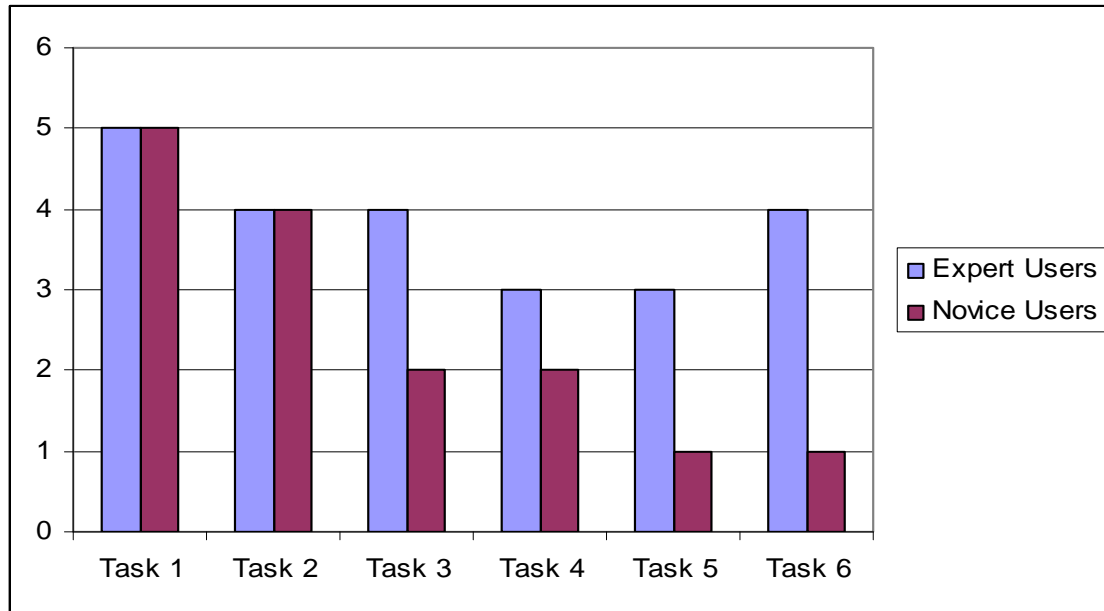


Figure 21. Usability Test Results

In the above chart, on the Y-axis we have the comfortable level of the users while performing the tasks and on X-axis the list of tasks performed. We analyzed that the expert users performed well in most of the tasks except the task related to models association. The reason behind the expert user's good performance is his prior experience with other IDEs. On the other hand, the novice users faced some difficulty while performing some of the tasks. The novice users got confused when they had to drag and drop the model over the controller component for interfacing. But once provided with a short training, the novice users performed better than the first time. Overall, the users appreciated the performance of our IDE over the other IDEs available. They also mentioned that the process of performing different tasks is simple and easy to learn using our StrutsHib IDE.

7. Conclusion

The goal of implementing a web-based application using Struts framework is achieved in this project. StrutsHib web-based application is user-friendly and it has features which help the user in the rapid development of their projects. It also takes the advantage of various new technologies such as: Ajax, Hibernate and jQuery.

This report focuses on the feature explanation, implementation details and IDE design decisions. It also discusses the advantage of a web-based IDE over a desktop-based IDE. In the end, the performance and usability test results clearly bring StrutsHib IDE into lime light. The users reported that the StrutsHib IDE is better than the traditional IDEs. However, at first the untrained users faced some difficulty using our IDE, but after a short training session and help section they performed well in the assigned tasks. The performance of our StrutsHib web-based IDE during the load process is much better than the desktop-based IDE. For all other operations, the performance of it is comparable to the desktop-based IDEs.

Currently, StrutsHib IDE is developed only to support Struts framework based applications. In the future, it can be extended to support other Java frameworks based on MVC architecture. StrutsHib IDE can also be enhanced to support team collaborations through integration with a source control system.

8. References

1. [Dionysios G. Synodinos] Web-based IDEs to become mainstream. Retrieved from - <http://www.infoq.com/news/2009/02/web-based-ide>
2. [Heroku] Heroku Documentation. Retrieved from - <http://docs.heroku.com/>
3. [Jingwen Ou] Aurorasdk. Retrieved from - <http://code.google.com/p/aurorasdk/>
4. [Jacob T. Biehl] FASTDash: A Visual Dashboard for Fostering Awareness in Software Teams. Retrieved from - <http://research.microsoft.com/pubs/64290/chi2007-fastdash.pdf>
5. [CO, Richelle Charmaine G.; OBALDO, Marc Anthony M.; ONG, Audrey Elaine G.] An Architecture for a Web-Based IDE. Retrieved from - http://netcentric.dlsu.edu.ph/CtrlSpace/DOC/MAIN/Main_Document.pdf
6. Struts. Retrieved from - <http://www.ibm.com/developerworks/library/j-struts/>
7. JQuery. Retrieved from - <http://jquery.com/>
8. Struts Tags. Retrieved from - <http://struts.apache.org/2.0.14/docs/tag-reference.html>
9. Walterzorn. Retrieved from - http://www.walterzorn/jsgraphics/jsgraphics_e.htm
10. Mozilla Skywriter or Bepin. Retrieved from - <http://mozillalabs.com/skywriter/>
11. Hibernate. Retrieved from - <http://www.hibernate.org/>
12. Hibernate Tutorial. Retrieved from - <http://www.hibernate-tutorial.com/index.html>
13. CKEditor. Retrieved from - <http://ckeditor.com/>
14. Development-as-a-service: Cloud9ide. Retrieved from - <http://cloud9ide.com/>

15. [Palvai, Tejasvi] Web-Based IDE for Interfacing View Controller. Retrieved from -
http://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1010&context=etd_project&sei-redir=1#search=web-based+IDE+research+paper
16. [Bhardwaj, Deepti] Work done for CS 297. Retrieved from -
<http://www.cs.sjsu.edu/faculty/pollett/masters/Semesters/Spring10/deepti/index.shtml?Bio.shtml>