CLS USER SERVICES WEB PORTAL*

D. Medrano#, L. Carter, D. Maxwell, CLS, Saskatoon, SK S7N 0X4, Canada

Abstract

The Canadian Light Source (CLS) User Services Web Portal is a collection of web applications that allows users and staff to manage experiment proposals, complete safety training and submit end-of-run surveys. Each user wanting beam time must submit a proposal describing their experiment. Once submitted, the proposal goes through a peer-review process where it is either approved or rejected. All on-site personnel are required to complete safety training. Staff and users are provided with training modules which are completed online. Most training modules consist of two parts: the presentation and the exam. The exams are graded automatically and the results are stored. At the end of each run, users are encouraged to complete an online survey. The survey gives users the opportunity to provide feedback on what was good about their CLS experience and what can be improved to provide them with better service. This paper will give an overview of the design, implementation and capabilities of web portal.

INTRODUCTION

The CLS is an international research facility with a large number and variety of visitors, including students, scientists and contractors. Access to the facility is based on three different roles: users, staff and contractors. Users are scientists that come from all over the world to run scientific experiments on the beamlines, collect data and publish results. Staff operate and maintain the facility and support users in their research. Contractors are personnel hired by the staff for a certain amount of time to complete a task. Contractors can range from labourers to project managers. The goal of the portal is to provide a user friendly and maintainable system to store information associating people with these roles.

PORTAL ARCHITECTURE

Figure 1 shows the architecture of the web portal. It consists of six parts. The first part is the Apache Tomcat servlet container. It is in charge of hosting the four web applications which are the main focus of this paper. These web applications include: training, proposal submission, end-of-run survery and proposal information Application Programming Interface (API). The second part is the Microsoft Active Directory (AD) server. Its purpose is to store all user, staff and contractor login information. Usernames and passwords are authenticated against AD when someone logs into the portal. All communication to AD is done through the Lightweight Directory Access Protocol (LDAP). The third part is the MySQL database server. It contains multiple databases for storing training, proposal, and end-of-run information. Java Database Connectivity (JDBC) is used to communicate with the server. The fourth part is the workflow engine. The workflow engine is used to create the peer-review process a proposal must go through once it is submitted. Workflows are created using the Yet Another Workflow Language (YAWL) [1] and communication is done through the Simple Object Access Protocol (SOAP) [2]. The fifth part is other web systems that make use of the proposal information API. Communication is done through HTTP using Representational State Transfer (REST) [3]. The last part is the web browser which an end-user uses to access the web portal.

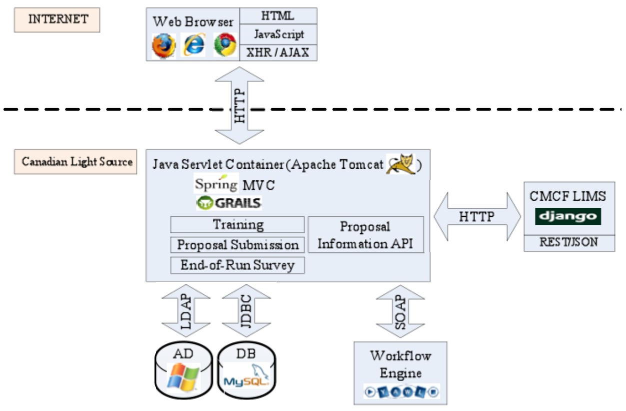

Figure 1: Architecture of the web portal.

TRAINING WEB APPLICATION

Any person coming to or working at the CLS that requires unescorted access must successfully complete and maintain their training. There is a standard set of training modules everyone must complete depending on their role. Users, staff and contractors use the web application to complete training and review training history. When someone takes an exam, a test is generated from a bank of questions in the database. Once a test is submitted, it is marked automatically by the system and the results are recorded. If a person fails the test they must retake it. Staff with the role of administrator use the application to create, edit or delete training modules, generate training reports, create new logins, manually enter training scores and manage training groups and roles.

The training application is written in Java using the Spring Model-View-Controller (MVC) [4] web framework. MVC is an architecture pattern that isolates domain logic from input and presentation. Figure 2 shows the breakdown of the application.

*Work supported by CANARIE under R&D project NEP-01 and performed at the CLS which is supported by NSERC, NRC, CIHR and the University of Saskatchewan.
#Dionisio.Medrano@lightsource.ca

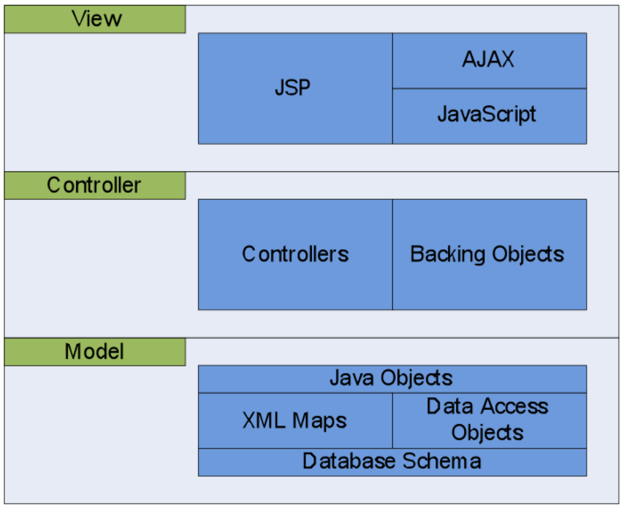

Figure 2: Structure of the training application.

The first layer is the ***Model***. The model is made up of the database schema, XML maps, data access objects, and Java domain objects. The database schema is implemented in MySQL. The domain objects are the Java representation of the database tables and they are mapped together by XML files called maps. This is done with a persistence framework called iBATIS [5]. The data access objects use the XML maps to fetch and store domain objects into the database.

The second layer is the ***Controller*** which contains controllers and backing objects. Controllers are small pieces of code that capture end-user input by handling HTTP requests and performing tasks. The backing objects are Java objects that do not fit in the model but help with getting data to and from the end-user.

The last layer is the ***View*** which consists of JavaServer Pages (JSP), JavaScript and AJAX. The JSPs use the JavaServer Pages Standard Tag Library (JSTL) to render Java objects and data in HTML. The JSPs also use JavaScript and AJAX to provide the end-user with a more rich and dynamic user interface (UI).

PROPOSAL SUBMISSION WEB APPLICATION

The proposal submission application allows users to create and submit proposals. When creating a new proposal, the user must fill out all required information. Examples of the information they must fill out include: type of proposal, name of proposal, funding sources, research team, scientific merit, beamline(s) they wish to use, safety hazards, and equipment they wish to bring. Once the proposal is submitted, reviewers are assigned. The proposal goes through a number of reviews. For example, a beamline scientist does a technical review to see if the experiment is feasible on the beamline. If the proposal passes all its reviews, a final grade is assigned and beam time is allocated. Administrators use the application to manage proposals, beamlines, endstations, equipment and cycles.

The proposal submission application is written in Groovy [6] and Grails [7]. Groovy is a dynamic object-oriented programming language which runs on the Java Virtual Machine (JVM). Grails is an open source web application framework which uses Groovy. The application is divided into three reusable Grails components called plugins. These plugins include clsDomain, clsAdmin and clsStaff. The clsDomain plugin contains the domain model, meaning that it has all the Groovy objects and relationships between them. The clsAdmin plugin contains controllers and views that administrators use. The clsStaff plugin has the controllers and views that staff and users use. Both the clsAdmin and clsStaff plugins use the domain model from the clsDomain plugin.

Grails uses Grails Object Relational Mapping (GORM) to generate the database schema and to map the database tables with the domain objects. This is done by Hibernate [8] which is an object-relational mapping library and persistence framework. Figure 3 shows the database schema generated by the GORM. All the views are done using GroovyServer Pages (GSP). GSPs are very similar to JSPs but use a different tag library.

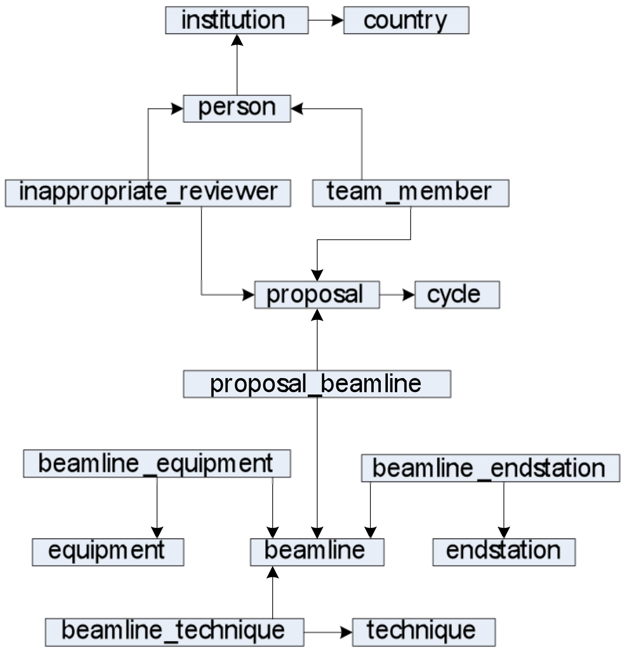

Figure 3: Database schema for the proposal submission application.

END-OF-RUN SURVEY WEB APPLICATION

In order to provide the CLS with feedback, users are encouraged to fill out an online survey about how well their research went while using the facility. The survey consists of a list questions split up into sections. Each question is multiple choice, but there are also some text fields where additional comments can be made. Once a user submits the survey, the feedback is shared with beamline scientists, the Director of Research and appropriate departments. This helps CLS staff identify areas where improvements can be made to provide users with better service. Administrators gather all the feedback to generate reports for the Users' Advisory Committee.

The end-of-run survey application is implemented using Spring MVC. It has the same structure as the training application. However, the domain model is very simple consisting only of two objects.

PROPOSAL INFORMATION WEB API

There are some other software systems at the CLS that are interested in having proposal information. The proposal submission web application already captures this information so it would be beneficial for it to share it instead of having each system duplicate and store the same information. In order for it to make the information accessible, a web API is being developed.

An example of another system that needs proposal information is the Canadian Macromolecular Crystallography Facility Laboratory Information Management System (CMCF LIMS) [9]. It is a web application for managing sample, shipping and experiment information related to macromolecular crystallography experiments at the CLS. It assists users to prepare crystal samples for shipping, request experiments to be performed automatically, manage experimental parameters, inspect and download experiment results.

When a user wants to do an experiment on the CMCF beamlines, they submit a proposal using the proposal submission web application. Once they have been allocated beam time they use the CMCF LIMS to manage their experiment. The LIMS requires certain information from the proposal, such as, proposal ID and primary contact person information. Instead of having the user enter that information again the LIMS uses the API to fetch it from the proposal submission database.

The proposal information API is implemented in Grails as a RESTful web service. The API uses the domain model from the clsDomain plugin but implements its own controller. When a system wants to fetch data, it sends an HTTP request to a resource backed by the controller. The controller handles HTTP request, looks up the information and formats it in JavaScript Object Notation (JSON) and sends it to the requestor. In order to protect sensitive data, the requestor is authenticated before the request is handled. Currently the API is read-only, meaning that data can only be read from and not written to the database.

SUMMARY

The design, implementation and capabilities of the web portal have been discussed at a very high level. Currently both the training and end-of-run survery web applications are being used in production. The end-of-run survey was deployed in January of 2008 and the training application in January of 2010. Both the proposal submission application and proposal information API are still in development.

REFERENCES

[1] http://www.yawlfoundation.org/
[2] http://www.w3.org/TR/soap/
[3] http://en.wikipedia.org/wiki/Representational_State_Transfer
[4] http://www.springsource.org/
[5] http://ibatis.apache.org/
[6] http://groovy.codehaus.org/
[7] http://www.grails.org/
[8] http://www.hibernate.org/
[9] M. Fodje, *et al.*, PCaPAC 2010 Conference Proceedings, THPL005