Andrew Chumich and Nathaniel Lane
Data Verity
June 17, 2014

Introduction

Client Description

In order to create a new front end for the Lon Capa online learning platform, we have been working with Data Verity, Inc., a company managed primarily by Gordon Flammer and Colorado School of Mines physics professor David Flammer. Most of their work involves selling their Enterprise Service Platform (ESP), an online database management system, to financial institutions to increase their productivity and efficiency. They do this by providing each client with an innovative, customized reporting system; offering expert, personal customer service; and delivering top level security and confidentiality.

Product Vision

Our goal in this project was to create a modern web application that will increase the productivity of Colorado School of Mines physics students and professors. The goal of any application is to have as few barriers between the user and the content as possible; the most difficult part of a physics student's experience should be the homework problems themselves, not the application used to view them. The application should provide a platform that allows students easy and fast access to their homework, lecture materials, and grades with an intuitive and modern user interface. We also want professors to be able to generate grade reports quickly; in the old system, this process could often take hours. Additionally, teaching assistants have to be able to hand grade essay responses in an efficient manner. The new system needs to be reliable, even under the strenuous conditions that often occur on the days following exams.

Requirements

High-Level Description

Our goal for this project is to create a bridge between CSM's Lon Capa and Data Verity's ESP. Currently, the CSM Physics department uses Lon Capa for online homework, in-class studio assignments, and online grade management. The student side of the program works quite well, albeit somewhat slowly. The course administration, however, suffers from poor performance, with grade generation often taking hours, and it implements an unintuitive interface for teaching assistants. Thus, there is a need to update the aging system. For this project, we are not creating a completely new platform; instead, we will be integrating functionality of Lon Capa with the new interface provided by ESP.

Functional Requirements

Teaching Interface Improvements

1. Improve speed when generating grading tables
2. Implement a more intuitive interface
3. Create a CSV uploader that will automatically store exam information produced by Scantrons

Rendering Lon Capa Questions

4. Display questions rendered by Lon Capa in ESP
5. Create an interface for Lon Capa question renderer to receive data and input from ESP

General User Content

6. Create charts and tables that allow students to track their progress
7. Display static content like PDFs, videos and webpages

Non-Functional Requirements

1. System performance will be improved from the original Lon Capa system
2. Use the Perl programming language to interface with Lon Capa
3. Use PHP and Javascript to interface with ESP
4. Data transfer between ESP and Lon Capa must be secure
5. System shall be testable throughout the development process
6. Version control will be handled by Subversion

Project Risks

1. It is possible that the complexity of the communication layer between ESP and Lon Capa exceeds that of creating a new system from scratch.
2. Neither of us has professional experience in web development.
3. We may overlook a security flaw that later causes issues when the system is used by real students and teachers.
4. PHP functions do not always translate perfectly to Perl, leading to longer development.

System Architecture

This project utilizes two pre-existing platforms: ESP and Lon Capa. Lon Capa is currently used by the Physics department as a teaching tool where students perform in-class exercises and do homework assignments. We are using its robust functionality for rendering and creating problems while removing dated interfaces and database structures. ESP is a web application framework and database management system written in ExtJS and produced by Data Verity. It already has functionality to manage users and user groups and was ideal for creating a new front end for Lon Capa.

Our first step was to create the communication layer so that we could call Lon Capa functions. With this, we could connect to ESP databases and write necessary information to them. This communication layer had to be written in Perl, since Lon Capa is written in Perl; this allowed us to directly call on Lon Capa and its submodules. Because of the way ESP uses post variables with PHP, we had to mimic the way it handles nested dictionaries using Perl. This proved to be a significant challenge due to limited functionality in Perl. Once it was in place, however, this submodule made it very easy to send and receive post variables.

With the communication layer done, we were able to create an application shell in ESP that allowed us to begin rendering questions from Lon Capa. This proved to be the most difficult aspect of the project due to Lon Capa's complexity. In order to render the simplest of problems, several environmental variables had to be set. The process of discovering these was quite time consuming and encompassed the majority of this project. Once we understood which variables needed to be set, such

as the path to the problem's XML data, we were able to begin to render questions using Lon Capa's xmlparse function. Once we were able to render questions, we were able to use MySQL databases to store problem and grading information.

With question data in the databases, were finally able to use ESP to construct the front end of the new Lon Capa system. The Learn application, which we constructed, on ESP uses Javascript to create a GUI and to call on functions in the communication layer to display a problem. The built-in functionality of ESP allowed us to create an intuitive tree structure for students and teachers to navigate through a course and to access its information. It also provides a simple way for teachers to set settings such as due dates and point values for questions.

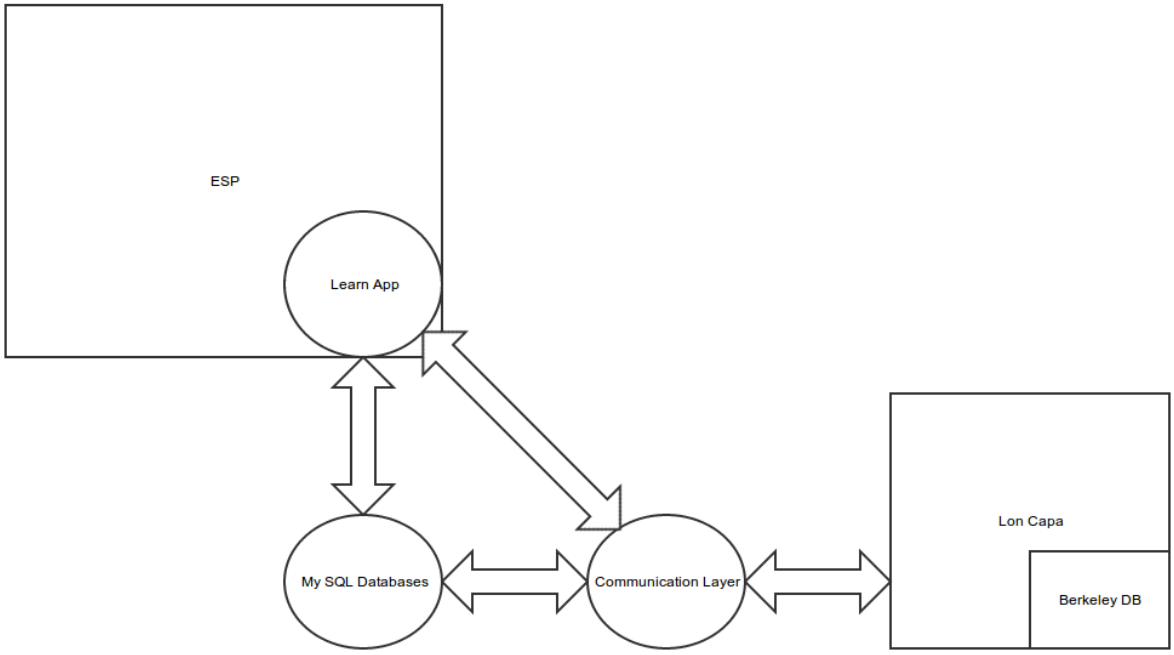

Figure 1: Diagram showing the system overview

Technical Design

Throughout this project, we made many technical design decisions. Some of these facilitated development while others slowed it down. For example, early on we were faced with the decision either to make a "master" session that was always logged into Lon Capa and had access to all questions or to take over the already present question-rendering functionality at a low enough level that user session variables were unnecessary. We decided on the latter as we expected that creating this session would take an unnecessary amount of time and that there would exist a single, self-contained function that rendered questions indepently from the user's environment. Unfortunately, this was not the case, as a vast number of environmental variables were required to be set in the rendering of a single question. By the time we realized we had made an incorrect choice in the matter, it was too late to revert that decision.

We also decided to implement an entirely new MySQL database system. This was based on the fact that Lon Capa uses Berkeley DB, which is fast for individual reads but slow for bulk access, to

store question and response data. On top of all this, the way it is implemented in Lon Capa requires a large number of joins and advanced operations in the process of generating class-wide grading tables.

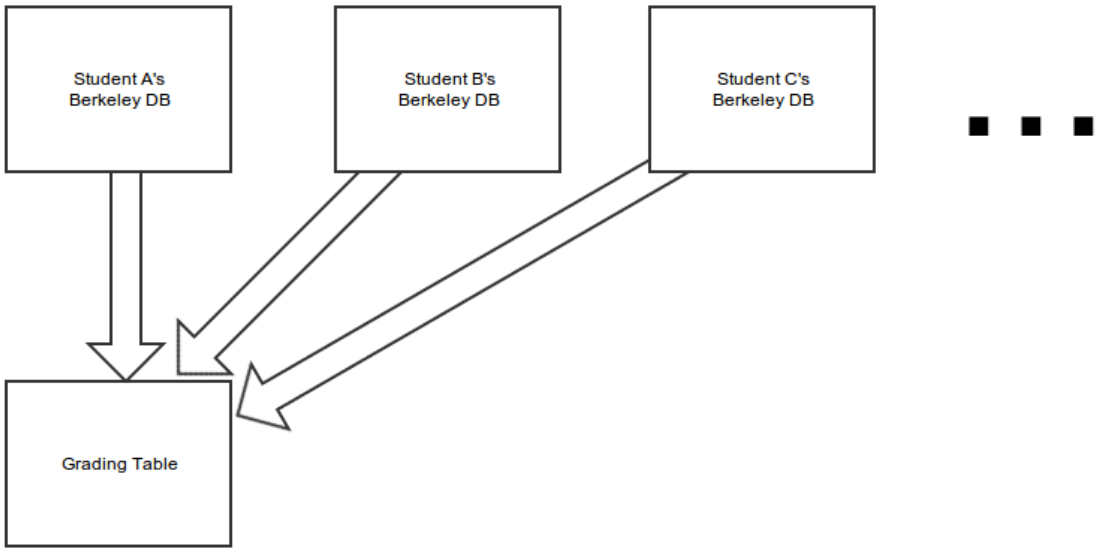

Figure 2: Lon Capa's grade generation relies on a DB structure that requires joining every student's grading table

While it was impossible to remove Lon Capa's dependence on its databases completely, by storing the renders of each problem in our system, we were able to reduce the dependency to one-time operations.

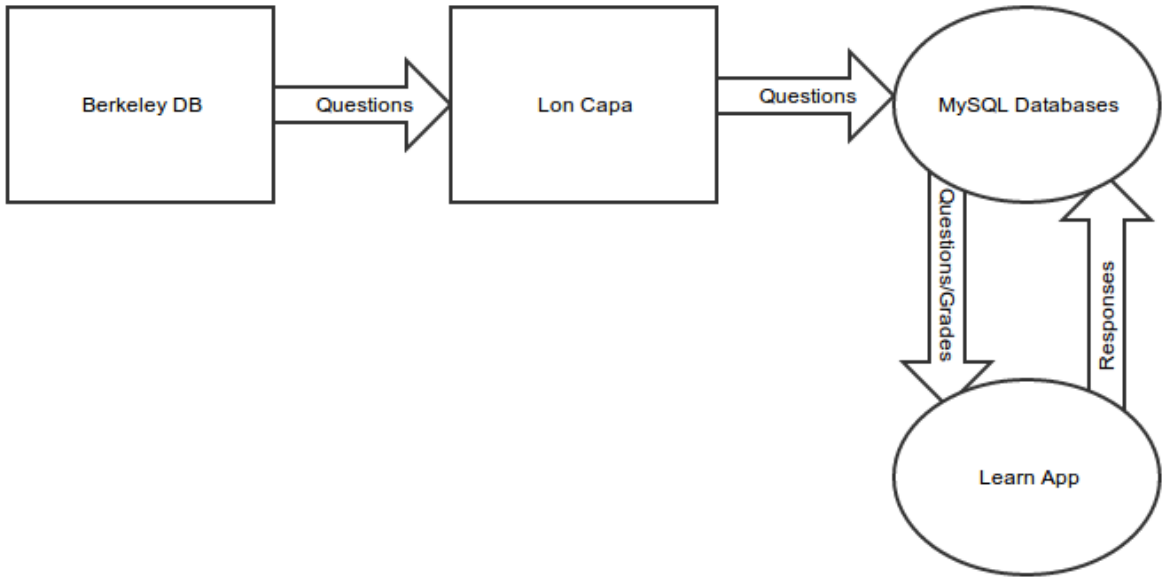

Figure 3: The new system only uses Berkeley DB once to retrieve question and answer data for a problem. Problem statuses are stored in MySQL exclusively. Grading information is stored in a single database.

Whenever the new system is installed, a course administrator may retrieve the contents of an old course quite simply by running a script called importCourseContents, which can be found in the Admin Manager in ESP. This script works by analyzing default.sequence, a file that contains the course

contents, and creates entries in the MySQL databases for its contents. This way, professors will not have to recreate entire courses and can keep all of their old questions.

Design Decisions

One of the earliest design decisions we made was to use Lon Capa's already present question-rendering abilities rather than the alternative, which would have been to write a tool that rendered the XML from Lon Capa's databases. Considering how well Lon Capa already does this, we saw no reason to rewrite this functionality. We believe that this decision proved to be the correct one even though Lon Capa makes rendering questions difficult. Due to the nature of the project, the majority of decisions were technical and did not pertain to design.

Lessons Learned

This project allowed us to gain valuable experience in working with large legacy code projects. One might naively approach this kind of project by attempting to understand the code in its entirety. With projects that contain hundreds of thousands of lines of code, this approach would prove fruitless. Through this project, we learned how to efficiently scan code to find relevant functionality.

Additionally, we learned how to properly seek help. If we had been in a situation where we could not ask an experienced programmer questions, we would not have been able to make significant progress on this project. That said, knowing how to ask proper questions is equally important; one must gather enough information about the issue in order to ask clear and informed questions. Simply asking "What do we do now?" is a waste of everyone's time and resources.

Results

For this project, we have successfully implemented a new database system with MySQL. This will facilitate faster load times and more efficient grade evaluations. We have also successfully created an interface by which we can display output from Lon Capa's question-rendering system. This uses Perl to capture the rendered page and uses AJAX to record student submissions in the new databases. Additionally, we have created a new graphical user interface (GUI). For teachers, this provides a more intuitive interface to specify question parameters and hand grade student essay responses. For students, speed has improved considerably as we are not reloading the entire page on each request.

This project had a broad scope, and some features were not implemented due to time constraints and lack of manpower. Originally, we wanted to include charts and tables that would allow students to easily track their progress; this has not yet been implemented. ESP has built-in charting tools, which will make implementing these in the future will be a relatively simple task. Additionally, we had planned on creating an interface by which teachers could easily upload exam grades with a CSV file

generated by a Scantron. This too has not been completed. Further, checking for the correct answer has not yet been implemented.

Appendix: Future Work

Due to the lack of man-hours, some features were not implemented, while others require some more work. This appendix discusses those features.

Question Rendering

At present, the new system is not capable of rendering complicated questions, such as the ones that have random number generation.

Handling Answers

There is currently no way to evaluate answers as being correct or incorrect. In general, this should be fairly simple; however, for formula responses, there must be a system that can ensure that the submission is mathematically equivalent to the correct answer. Further, once students provide correct answers, the system needs to inform them that they are correct and stop accepting submissions from them.

Simple Content

At present, there is no rendering of simple content such as PDF's and text documents. This should be added as another case in the handler in Learn.pm.

Problem Creation

There is currently no way for a course administrator to create new questions in the new system.

Charts and Graphs

There are currently no charts or graphs implemented for students to track their progress in the course.

CSV Uploader

There is no existing functionality to create exams and process them with a Scantron.

Security

Though some measures of security have already been implemented, they are not extensive enough at the moment to warrant full release of the system.