Integrating Marratech video conferencing with the Moodle course management system

Mustafa Paksoy (author), Paul Bible, Sam Cook, Chris Dragga, Sutha Ganesharajah, Tobias O'Leary, Daniel Potkalesky, Zain Shamsi, Amit Shresta, Mike Treder

National Institute for Technology and Liberal Education, Southwestern University, Georgetown, TX

Introduction

Course Management Systems (CMS) are instrumental to collaborative projects between institutions of higher education. There currently exists several full featured CMS's and video conferencing applications. However, these applications are not integrated.

Our project over the summer was integrating a popular CMS, Moodle, with the powerful video conferencing suite Marratech. The resulting applications combines the strengths of these two components and provides the user the best of both worlds.

Two Components

Moodle
- Open source
- Runs on LAMP (Linux Apache MySQL Php)
- Extensible with course activity modules

Marratech
- Proprietary
- Runs on Java Enterprise edition
- Extensible through the API

Design

Simplicity: Video conferencing sessions are scheduled within Moodle like any other event.

Drop-in installation: Our application does not require Moodle or Marratech to be modified in any way.

Configurable: Administrator can change, among other things, max. bandwith usage, contents of email notifications sent, how strictly meeting durations are enforced.

Tightly integrated: Users can join meetings directly from Moodle. Meeting rooms are provisioned automatically when the meeting is started. User accounts, access permissions from Moodle are automatically carried over to Marratech. The administrator can also sync the Moodle module and the Marratech manager with a single click.

Application structure:

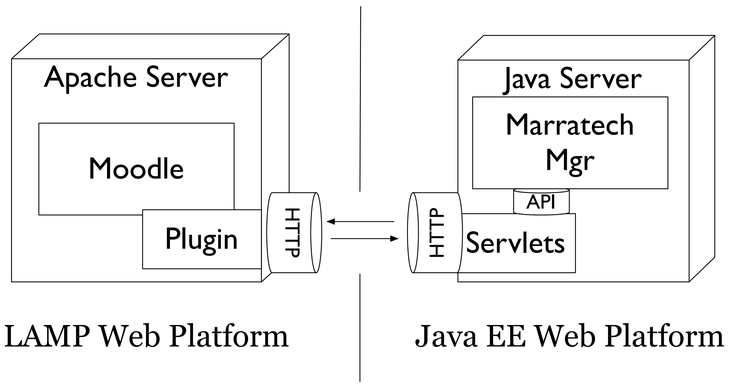

Figure 1. High level application design. Two main components of our system run on separate web platforms. Our Moodle plugin communicates over HTTP to Java servlets. Java servlets access the Marratech Manager over the API.

Methodology

During our software engineering project, we followed an agile software development methodology called Extreme Programming (XP).

Collective code ownership: Team members do not specialize. Everyone is responsible for all parts of code.

Incremental design: Development process is divided up into incremental releases. At each release the design evolves and is continually revised. This allows the customer to see the product in development and steer the project to their liking.

Test driven development: Unit tests are written before any piece of functionality is added. These tests are written in a easily repeatable way using utilities such as JUnit. As more functionality is added, code is continually integrated and tests allow product to remain stable.

Acknowledgments

We thank: Eric Jansson and Eric Harper for supervising our project and providing crucial input; our instructors for the first half of the project for sharing their knowledge and experience with us; NITLE staff at Southwester Uni; and Marratech AB for their assistance with the API. This project was funded by NITLE.

For more information contact Mustafa Paksoy
mpaksoy1@swarthmore.edu

NITLE