E-Grade

Top-Level Design Document

author: Mark Humphrey <msh>

date: February 21, 2003

---

1. Project Description

Every year TAs are confronted with the problem of how to store and calculate grades in some sort of organized manner. Often the solution is a set of patched together and broken scripts which are difficult to use and generally ineffective. The purpose of this program would be to provide a tool which would allow the easy administration of grades by TAs as well as the capability for students to view their grades. This will be accomplished through the use of several different user interfaces: web, gui, and command line. The ideal is to provide a flexible of a system as possible, so that it can meet the varying needs of many classes.

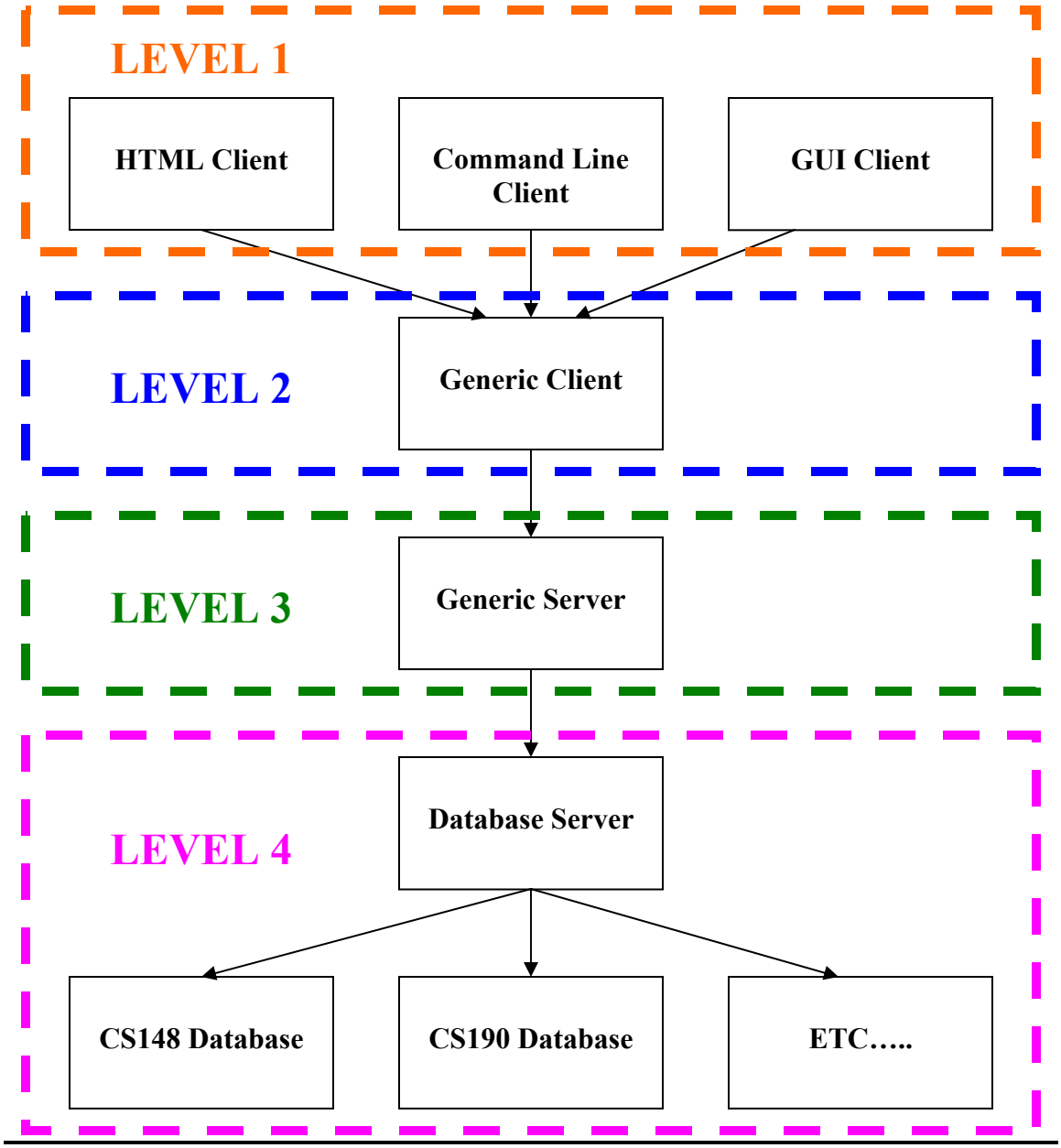
2. Specification

This project has no particular specifications document but instead takes aspects from several documents on the topics of grading, teaching and online organization. This project takes a minimal set of these features from each and combines them into some new.

3. Components Overview

The diagram on the next page shows the main components of the project and their dependencies. There are four levels total, however, the fourth level is handled by an external package and is only shown for clarity of design. As a result there are five separate components, each designed so as to be coded by a single programmer. Each component is not dependent on any of the components in the same level and is only dependent on the one component in the higher level. This is accomplished through the use of queries which return information as the primary means of communication between components.

4. Levelized High-Level Component Diagram



5. Component Description

Level 1:

HTML Client

Provides a web interface to the grading system. This would most likely have a limited set of features which would primarily consist of the ability to view grades, both by TAs and students, as opposed to actually editing them. While the ability to edit grades could be considered, html generally provides a weak interface with which to do so, especially on a large scale. It will send queries to the "Generic Client" for all information requests.

Command Line Client

Provides a Linux-style command line interface to the grading system. The main purpose of this would be to allow the user to interact with the program through the use of scripts, giving the system the ability to more effectively process large repetitive tasks. This would also make the program more flexible allowing the particular user to bend it to his or her particular needs. It will send queries to the "Generic Client" for all information requests.

GUI Client

Provides an intuitive and easy to use graphical user interface to the grading system. This interface is primarily for administrative purposes and allows easy graphical manipulation of data and statistics. The primary motivation for its existence is the feeling that html lacks the flexibility and power to effectively graphically display and manipulate large sets of data. It will send queries to the "Generic Client" for all information requests.

Level 2:

Generic Client

Provides a comprehensive client which provides a layer of abstraction between the specific clients and the server. It has extensive methods for returning any piece of information which the database holds, as well as passing on database modification requests. It sends queries to the "Generic Server" for all information retrieval and modification requests.

Level 3:

Generic Server
Provides an interface which takes the clients queries for information retrieval and modification and translates them into the appropriate calls to the sql api. This will involve some simple logic to assure that the proper databases are updated and all related information is kept up-to-date. Receives queries from the "Generic Client" and

Level 4:

Note: This level is handled by sql, but is included to get a complete feeling of the project design.

Database Server
Provides an interface with which to modify and retrieve information directly from the databases.

Databases
Each database is of some yet to be defined format and provides a compact and organized way in which data is stored and retrieved.

6. External Dependencies

This project requires the acquisition of a computer for use as a web and sql server. It also relies on someone successfully installing and setting up these two servers. The setup and use of a secure authentication method for users is also another external dependency. In addition use of the sql api and cgi will also be required.

7. Group Breakdown

Note: Numbers indicate number of group members. Also, no particular people were assigned to individual responsibilities as it seems silly to do so not even knowing who is in the group, let alone what people's individual preferences are.

Project Manager (1):
The leader of the team, must be both responsible and feel motivated by the project. Works to keep everyone on schedule and task. Arranges and leads meetings and just generally keeps the flow of communication open. Constantly reviews the progress of the project and jumps in to assist in any tasks which may be lagging.

Librarian (1):
Keeps track of all paperwork and basic records of group meetings. Maintains the rather primitive project web page and writes the documentation for the installation and use of the project.

Support Technician(1):
Handles many of the tedious and time consuming tasks such as creating makefilesand setting up CVS. Also will handle the setup up of the sql and apache servers as well as any support which needs to be provided with either.

GUI Designer(1):
Graphically designs the layout of all three of the user interfaces. Works with the component programmers to assure that user interfaces are intuitive and aesthetically pleasing.

Primary Tester(1):
Works with the individual programmers to develop and implement testing plans, both for individual components as well as total integrated project. This may require writing some simple test drivers.

Component Programmers (5):
The primary coders in the group, they are each responsible for the competition and design of a particular component in the project. They are expected to provide detailed internal designs as well as interfaces for their components. Incremental testing and debugging is also a responsibility.

8. Schedule

- ***2/28*** *Top-level Design Selection* :
    - Project manager chosen
    - Project librarian chosen
- ***3/3*** *Initial Group Meeting:*
    - Individual component responsibility assigned
- ***3/4*** *User Input* :
    - Question possible users on overall interface
    - Incorporate any resulting changes
- ***3/9*** *Final Top-level Design* :
    - Revisions based on group feedback completed

- Assignments of group member roles finalized

- ***3/12*** *Interface Proposals* :
  - Each person in charge of a component of the system should produce a proposal as to their external interfaces.
  - Proposals made available both electronically and in hardcopy and should be coordinated by the project librarian.
- ***3/17*** *Interface Comments* :
  - Between the last class and this one, each student should go over all the interface proposals that affect their components of the system and provide feedback, comments and suggestions.
  - The project manager and librarian should go over all interfaces and provide feedback.
- ***3/21*** *Final Interface Definition* :
  - Each person should have fixed their proposed interfaces by this date. This means that they have to respond to all comments and to get final approval for their interfaces from the project manager.
  - The project manager is responsible for insuring that all interfaces are consistent and complete. Changes to the interface after this date must go through a full review process including in-class justification.
- ***4/1*** *Support Tools Completed* :
  - Computer installed and setup
  - Web server installed an configured
  - SQL server installed and configured
- ***4/4*** *Detailed Designs* :
  - By this date each individual should have finished and had the detailed design for their components approved. If a full-design approach is being taken this means that they should have written modspecs for all their modules and have gone over these with both the grader and with the project manager.
  - Once approved, the design should be submitted to the project librarian.
  - If a prototyping approach is being used, then one or more prototypes should have been completed by this date and the actual implementation should be decided on by the TA and project manager based on experiences with the prototypes.
  - Once the prototype is settled, the individual is still expected to do full modspecs for submission to the project librarian within a reasonable amount of time. :
- ***4/10*** *Progress Meeting* :

- Discuss what current status of components is so far
- Make sure everyone is on track
- Address any problems or surprises

- ***4/16*** *Initial System Integration* :
  - This is the magic date where we should see the initial version of the projects all fit together and work.
  - This should contain the overall control structure and most of the user interface, but specific functions may be stubbed. All the essential components should be here in some form.
- ***4/27*** *Full System Implementation* :
  - This is the target date for having a fully implemented, functional basic system. The system will be demoed in class. Some bugs may still exist, but system should be stable enough to demo.
  - Functionality will be frozen at this point
- ***4/27-5/2*** *Testing and Stabilizing* :
  - Iron out any bugs and stabilize code
  - Finalize any user documentation
- ***5/2*** *Public Demo:*
  - Projects will be demoed to the public in Lubrano on this afternoon.
- ***5/9*** *System Submission* :
  - This is the final date for submitting the system. The system should be fully functional and installed for users with full documentation available.
  - Full listing of the complete system as well as the final project notebook handed in.
  - Full demonstration of system for the professor and TAs.

9. Assumptions Made

Since this project has not yet really been defined, most of the details have been pieced together from several of the projects. This is a fairly significant weakness as there are no firm requirements or specifications for this particular project yet. However, most of the proposed projects are a little fuzzy on the details so this could be overcome with some extra effort. In addition, it is assumed that we can devise some sort of system that is general enough that it can fit any class's needs. When one really thinks about the different grading systems and criteria that exist it is actually a very difficult task. The variety tools which are used in this project may also prove a difficult obstacle. However, I think the design and implementation of this project is definitely a good software engineering exercise.