Joseph Fuqua
jfuqua
CS190

Top-Level Design for Bear/Advisor

Description of Components of Design

User Interface/Browser – This is the web browser that the user views the HTML interface with. It is responsible for displaying the pages the user will see and interact with. It must get the page to display from the Component interface and return any user input to the Component Interface.

Component Interface – This component is the interface between the user input and the data processing aspects of the program. It defines the interactions between the front and back ends of the program. By making templates for each web page with set options for user interactions all the possible interactions can be encapsulated into methods that this interface can understand. This prevents the need for each component to have knowledge of the UI and vice versa. Having predefined all the functions of each component and their related web pages further allows the interface to simply treat them as processing modules and eliminate the need for the components to have knowledge of the interface. The output returned by the component's methods will then be sent to the HTML Factory to be made into user-ready web pages and these pages sent to the interface for display.

HTML Factory – This component creates all the actual web pages that the user will see. It has templates for each of the web pages the user will need to see and will be able to create these through its member methods. The parameters of these methods will account for the changing content of the templates. The methods will return a ready-to-display web page.

Add Course Permissions – This component handles accepting and denying students to a limited enrollment class. Its methods will handle the relevant data from the Component Interface and will make changes to the registration data through the Database Administration Interface. Success or failure will be reported to the Component Interface through the return value of the methods.

Registration – This component handles all the registration activities for students. Adding and dropping a class, changing a grade option, changing a section, deciding on audit/credit, etc. It has methods to handle every action the user could choose on the Registration web page. These methods access and modify the relevant data through the Database Administration Interface and report back to the Component Interface through return values.

Login – This component handles logging in and out of the system. It checks the user name and password for validity and chooses the options available based on the user's ID (professor, student or administrator).

Smart Search – This component handles the duties of the Advisor Smart Search. Given the input data it will query the databases for classes matching the criteria and return any results via a return value.

Course Review – This component takes care of adding course reviews to the review database via the Database Administration Interface. It reports the success or failure to the user via the return value of the method called by the Component Interface.

Database Administration Interface – This component is another interface that encapsulates and defines all interactions between the components and the data. This allows components like Registration, Permissions and Smart Search to have access to all the data without circular dependencies.

The remaining entities on the graph are databases that are accessed and updated by the other components and interfaces.

External Dependencies

This whole project is based on contacting the user through the Internet and thus is dependent on that system and the user's Internet Browser to be successful. We will have to also implement some database software to organize the data for classes, reviews, students, etc. and we will be reliant on that system. Making this system client/server will also add a network dependency.

Task Breakdown and Group Organization

Project Manager – This person will be responsible for the overall progress of the project. This means keeping the team on track in terms of schedule and direction. The Project Manager will be called upon to resolve any personnel disputes not related to code and to check on the progress of the individuals and help to keep them up to speed with their colleagues. The Project Manager is the big-picture person the pragmatist that keeps to project moving ahead.

Architect – the architect is responsible for the design of the program interactions. This person will know the design intimately and will be able to offer suggestions about how to implement pieces of it. The architect is the keeper of the concept; he is in charge of designing the project and seeing that that design is kept to. The architect must oversee development to ensure that the design is kept to so the other team members can work simultaneously.

User Interface Modeler – This person is responsible for the look and feel of the program. Since this is a user-end application and will be marketed to many barely-computer-literate students it must be intuitive and easy-to-use. The job of the modeler will be to decide what to place in front of the user and where. This person will not be directly responsible for writing the code to generate these screens but will work with the Programmers to make sure that the user interfaces are created properly. In terms of this project this person will be most directly involved in the development of the HTML Factory and will speed the implementation of it by creating detailed specifications for each screen to be written.

Lead Programmer (Surgeon) – This person is responsible for most of the actual code generated in the implementation of the project. The lead programmer is the one who takes the architect's design and makes it concrete. He will be responsible for the intricacies of the code and will know all of it intimately.

Assistant Programmer (Co-Pilot) - The Assistant Programmer is also involved in coding. He acts to help the Lead to develop code faster by taking over some of the actual implementation. He will also know the code intimately but will not be responsible for making decisions on implementation.

Toolsmith – This person is the other that will be involved in code generation. His primary role will be to implement subroutines to take some work off of the Programmers. This will only involve writing small, independent subroutines and methods but will be very useful due to the number of these that will exist in implementing the interface interactions.

Language Lawyer – This person is the language expert. C++, Java, HTML, SQL or whatever, this person will need to know the details of the reserved words and the intricacies of the language. This will help tremendously in debugging the components and the whole project since there will have to be multiple languages used and their interactions could be difficult.

Tester – This person will be responsible for designing tests and testing individual components and integrated systems. This will be made easier by the interface systems and these systems will allow the tester to begin testing right away. He will work with the Programmers and Architect to find the problem in the system. He will also work with end-user testers and get their input.

* A note on the use of the male pronoun – I realize that there are three women in my group but I still used the male pronoun (“he”) in describing some of the duties. It is simple, clear and concise. Nothing else is.

<u>Schedule</u>

February 25 - Top-level design’s handed in
March 3 - Top-level designs discussed by group
March 10 - Revised top-level design completed and frozen
March 15 - Interfaces for component interactions proposed
March 20 - Interfaces reviewed by group members and discussed
March 24 - Revised interfaces agreed upon by group
March 25 – April 2 – Spring Break
April 5 - Detailed designs for individual components finished
April 10 - Individual component testing begins
April 15 - Components debugged for initial integration
April 17 - Initial System Integration
April 20 - Integrated system testing
April 24 - Initial integrated system debugged.
- Additional functionality added
April 28 - Full System Implementation
- Functionality frozen
May 1 - User Testing and debugging

May 4	- Public Demo
May 10	- Code freeze
	- Final Documentation of code and known bugs
May 12	- Final submission