Teaching Tools

Top-Level Design

February 21, 2003

Emma Boroson (eboroson)

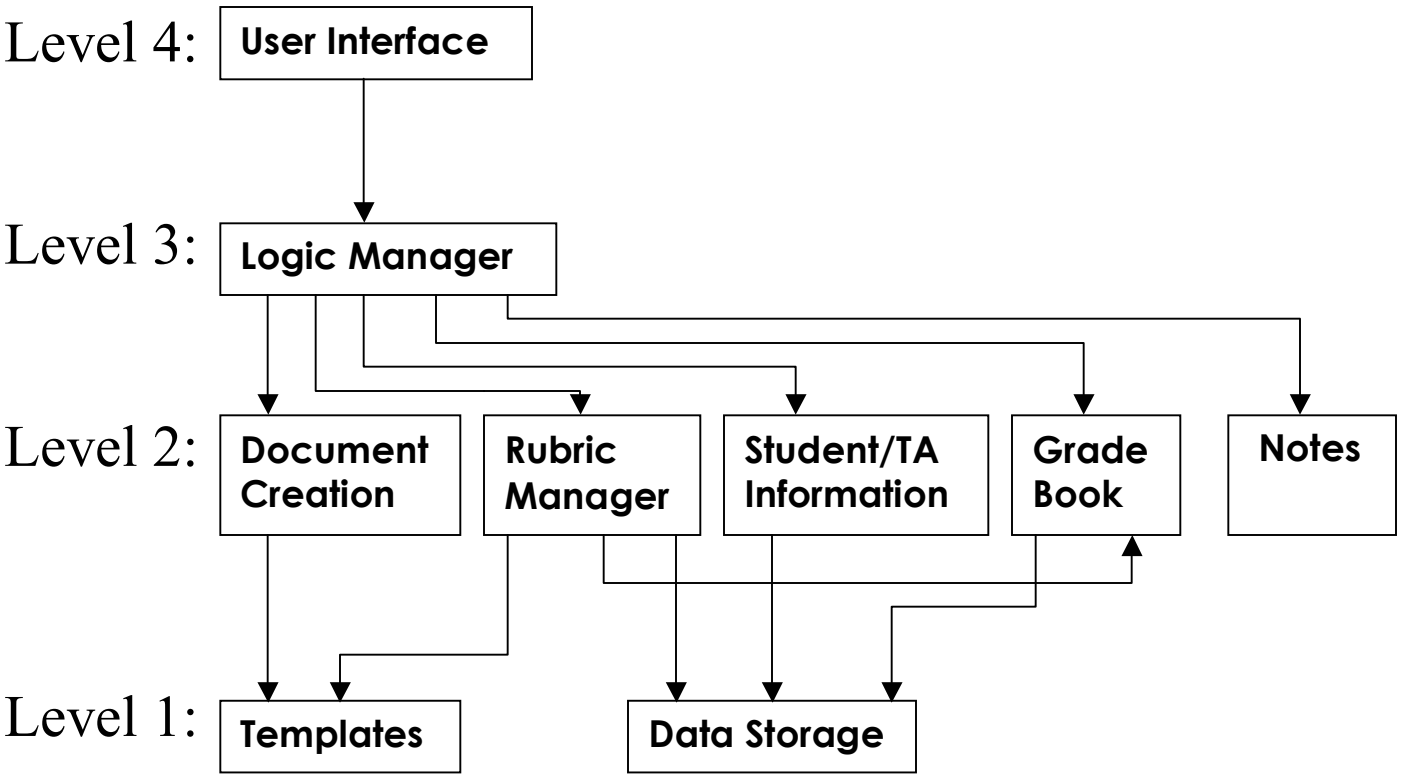
Introduction

This top-level design document is based on the Teaching Tools specifications document of lmolinar, which can be found at http://www.cs.brown.edu/courses/cs190/asgns/2-7/tmp/lmolinar.pdf

The main chance in functionality of this particular program is the deletion of the direct E-mail capabilities of the original specifications. (More details can be found in the component description section following.)

Please see the following page for component diagram.

Levelized High-Level Component Diagram



Component Descriptions

User Interface:
The main GUI will essentially be the main button panel, which gives the option to directly get to any one of the five "subdirectories" (Grade Book,

Document Creation, etc) plus the subscreen, which will show either the login or the chosen "subdirectory". (please see original Specifications Document for clarification)

- Subscreens:
  - Login
  - Grade Book
  - Document Creation
  - Student/TA Info
  - Rubric Creator
  - Notes

The main GUI (Level 4) should know which subscreen is being used at any given time, and know how to handle user input from any screen (know which part of the main logic to call)

Logic Manager:
This is called by the User Interface and is handed data/instructions to give to one of the five Level 2 modules – sort of acts as an interface between the GUI and all the "subdirectory" parts of the program.

Document Creation:
An editor-type function in which to create lessons, lecture notes, etc – as per Specifications Document. This module may call upon the Templates module – may receive a pre-set-up modifiable file that the user may add to and modify.

Rubric Manager:
Also similar to Document Creation – may call upon Templates in a similar manner. The user has the option of setting up a connection between the Rubric Manager and the Grade Book, where grades instantiated in the Rubric will be automatically entered into the Grade Book.

Student/TA Info:
Similar to Grade Book layout – takes information from the user (through the User Interface and then through the Logic Manager) and stores given info.

Grade Book:
A grid-like layout in which to store, add, and modify grades – as per Specifications Document. This will take information from the Logic Manager and add or modify grades accordingly.

Notes:
An editor-type set-up in which to create notifications for some or all users. This *differs* from the initial Specifications Document in that there is no direct way to E-mail from this program. It is possible to set up E-mail *lists* and contacts, and also possible to *write* E-mails. However, sending the mail must be done through a separate mail client.

Templates:
These templates may be used by the Document Creation module (with given lesson set-ups, etc) and may be modified by the user. There will also be templates available specifically for creating a rubric (to be used by the Rubric Manager.)

Data Storage:
This will be a database of information called upon by the Grade Book and Student/TA Info. Information can be inputted to storage through these two given modules.

External Dependencies
There are no external dependencies in this program.

Task Breakdown and Group Organization

Manager: This person is in charge of everything having to do with organization of the group – scheduling meetings, checking up on everyone's progress, encouraging, punishing (hopefully that won't be necessary!), making final decisions if disuptes arise – this person, in the end, is responsible for getting the project completed on time.

Documentation/Librarian: This person must know the functionality of every part of the program at any given time – must keep records of each aspect of the project and must update the records as progress and changes are made – is responsible for writing the online user documentation, so should know the ins and outs of the program.

Head Architect: This person is in charge of the final design of the program. While all group members will help in the design, the head architect makes final design decisions if disagreements occur. The head architect must know all interfaces and interactions in the program and must be able to answer questions from all programmers at any time throughout the implementation process.

Head Tester: This person is, in the end, responsible for making sure all parts of the program have been thoroughly tested (including integration testing) – this may mean that the head tester comes up with and implements all the testing, or it may mean that the head tester simply makes *sure* that each programmer has sufficiently tested their part of the program.

Programmers: To be split up as follows:

GUIs
Logic Manager/Interfaces
Templates/DataStorage database stuff
GradeBook/ Student/TA info
Document creation
Rubric Manager
Notes

} some combination of 2 and 1 of these

Proposed Schedule

2/28: group positions selected, top-level design selected

3/9: Final top-level design – revisions completed -

3/12: Interface proposals by each person in charge of a programming component – should be coordinated by group librarian

3/17: Interface comments and feedback by all group members – all have reviewed and discussed proposed interfaces – Group manager, librarian,a nd architect should decide on changes based on feedback

3/21: Final interface definition for each component – all must be approved by project manager and architect to make be sure the interfaces are complete and correct

4/4: detailed designs for each component – coding should have begun by this point -- full design proposed to head architect/librarian

4/4 – 4/16: INTENSE CODING TIME – a week and a half for all component programmers to get basic functionality up and running – this includes component testing, overseen by head tester

4/16: initial system integration: specific functions need not all be completed – but some aspect of all parts of program should be working together – basic integration testing at this point

4/27: full system implementation: fully functional – may still be bugs – but working enough to demo in class – Manager should make sure all is in order for final demo to be completed

5/2: public demo

5/2 – 5/9: final bug-checking and fixing – only minor details should be worked on at this point – Head tester doing final checks, Librarian putting final touches on user documentation

5/9: System Submission **– ALL IS COMPLETE!**

Assumptions

The intial specifications document is rather clear in all aspects, so I didn't have to assume much. The change I did make to the Notes module is minor and I chose to do this to simplify the breadth of this project – if E-mail were involved, there would have to be an entirely new compenent involving a mail client and networking. This way, I kept nearly all of the functionality of the original program without complicating it.