Teaching Tools
Top-Level Design

February 20, 2003
By Liza Molinari (lmolinar)

PROJECT DESCRIPTION

As explained in the Specifications document, Teaching Tools is a stand-alone program that centralizes all course information. Grades, lecture notes, assignments and all course-related materials can be stored, managed and classified. The target users for Teaching Tools are professors and teaching assistants at any academic institution. Users are not assumed to be very computer literate. Teaching Tools will be designed to be extremely user-friendly.

This project has not been altered since the Specifications stage to include features from other teaching program ideas. The core components of the program are unchanged: grade book, document creation, rubric manager, notes and student & TA information.

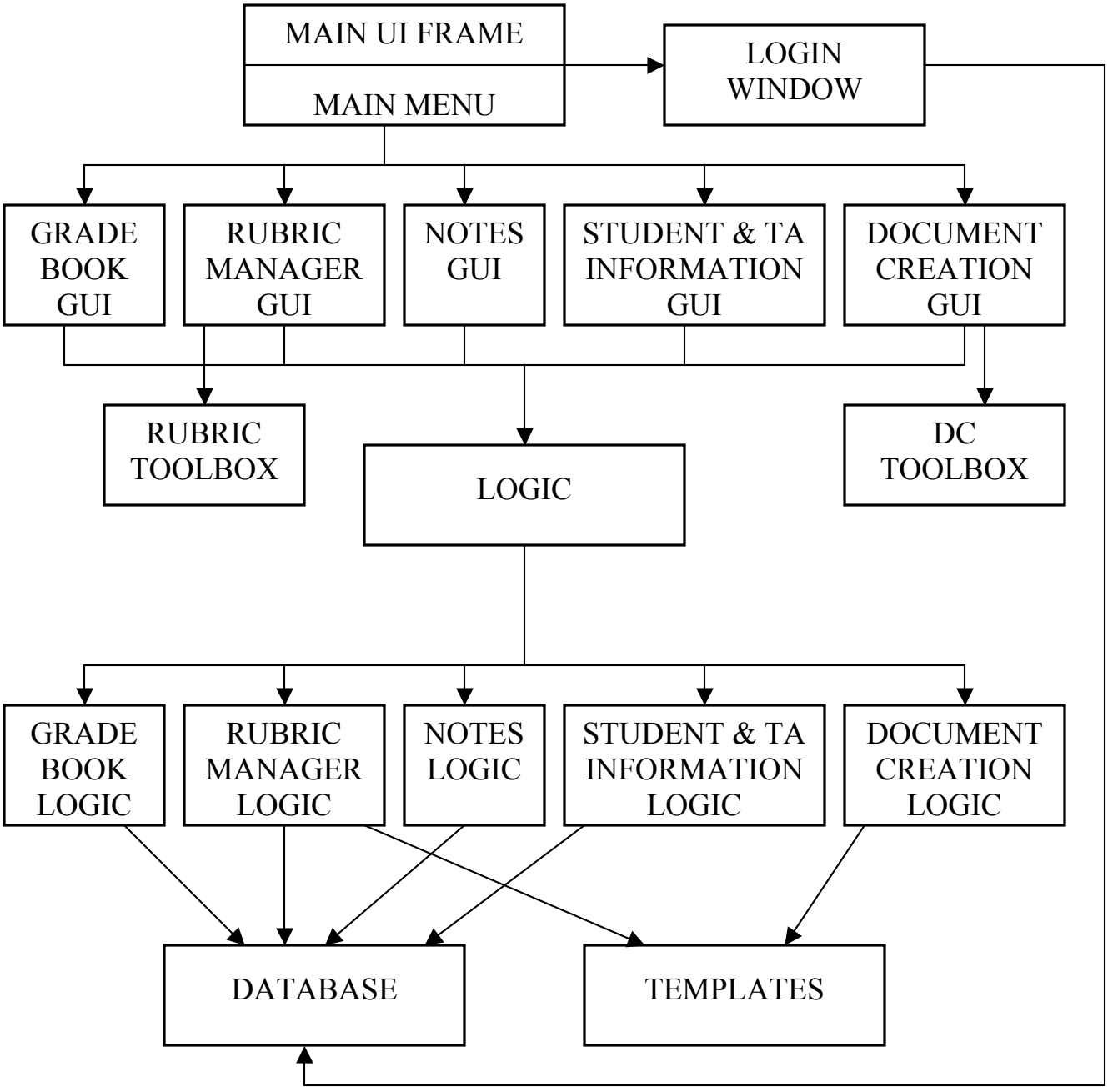
DESIGN OVERVIEW

The following document contains a top-level design for Teaching Tools. It will describe and diagram components and their interaction. It will also explain individual roles within the group, a project schedule, assumptions and external dependencies.

COMPONENTS OVERVIEW

There are 17 components in the top-level design; however, they vary in size and complexity. The core areas of Teaching Tools are Grade Book, Rubric Manager, Notes, Student & TA Information, and Document Creation. As such, each has a GUI and Logic component. The GUI component is the main work window for that section of the program (described in the Specifications, see diagrams at the end of the document.) The Logic component handles all the functionality of the corresponding program section. For example, the grade book logic component handles communication with the database, including inputting and retrieving grades. The following is a diagram of component interaction. Specific descriptions of individual components follow.

COMPONENT DIAGRAM



The above 17 components can be classified into 5 groups: Main User Interface, Core Component GUIs, Main Logic, Core Component Logics, and Data Components. The Main User Interface group includes Main UI Frame/Main Menu and the Login Window. Core Component GUIs include the above GUIs for the core components, the Rubric Manager Toolbox and the Document Creation Toolbox. Main Logic simply includes the

Logic. Core Component Logics include the above Logics for the core components. Data Components include the main Database and Templates used by the Document Creation Logic and Rubric Manager Logic.

INDIVIDUAL COMPONENT DESCRIPTIONS

Main User Interface

Main UI Frame/ Main Menu: As shown in Diagram A in the Visuals section (end of document, taken from specifications) the Main User Interface consists of a Main Menu and a Work Window. This class is the frame for the GUI and the Main Menu. The Main Menu simply displays a column of buttons. There is a button corresponding to each of the program's core components. When a user clicks on one of the buttons, the Work Window displays the Core Component GUI for the component corresponding to the button. When the program starts up and no button has been pressed, the Login Window appears in the Work Window. This also appears in Diagram A.

Login Window: The Login Window only appears when Teaching Tools first starts up. Its function is to validate users and direct them to the appropriate course data. It has a place for users to enter their user name and a password. It also asks new users to register. Once the user inputs their user name and password, the login window communicates with the Database to verify the input and grant permission for the user to enter the program.

Core Component GUIs

Note about all Core Component GUIs: The GUI does not carry out any functionality. It simply recognizes user input and passes it along to the corresponding Logic component through Main Logic.

Grade Book GUI: The Grade Book GUI is the work window for the Grade Book; it is shown in Diagram B in the Visuals section of the document. The user interacts with the GB GUI through pull-down menus, links and buttons It should be noted that there are three main GUI windows, the viewer, the modification window and the grade entry window. All three windows are very similar. Diagram B shows the viewing window. Clicking Modify Grade Book in the viewing window accesses the modification window. This window allows user to add/remove, edit and move columns. The grade entry window is accessible by the pull down menu in the viewer. It allows users to input grades (into text input boxes) in the appropriate assignment column. When users click Update, the grades are entered (through GB Logic) and they return to the Viewer.

Rubric Manager GUI: The Rubric Manager GUI is responsible for allowing users to create and fill in rubrics. It must also be able to send a representation of the current rubric to the Logic for File I/O. There are two main windows: one for creation and one for grading. The creation window has a toolbox full of boxes and tools that users can click and drag onto their rubric.

Rubric Toolbox: This toolbox contains items needed for creating rubrics. It will include a basic text box, an input box and a calculator box. There will also be templates available to the user. The templates specify text and box style. The calculator box will contain a value calculated from input boxes. All items are placed on the rubric by clicking and dragging from the toolbox. When an input or calculator box is added options will appear asking the user if they would like to connect that box to the grade book. If they choose yes, a grade entered in the rubric will also be entered in the appropriate column of the grade book.

Notes GUI: This GUI appears in diagram C. Its description is very similar to that of the Grade Book and Rubric Manager, only less complex.

Student & TA Information GUI: Visually, the Student & TA Information GUI is very similar to that of the Grade Book. Users can add columns/rows to the table in order to add information. They can also sort; however, this functionality is carried out in the S&T Information Logic component. The GUI merely accepts input.

Document Creation GUI: The document creation GUI is very similar to the Rubric Manager's GUI. There is a toolbox that contains the same options, with the exception of input boxes. There are also templates that the user can utilize for style. The main difference is that documents must be saved in a format recognizable by outside software, such as pdf format. They must also be printable.

Document Creation Toolbox: This toolbox functions the same way as the Rubric Manager's. It contains text and template options.

Main Logic

Logic: This Logic class is essentially a proxy that the core component GUIs use to communicate to the corresponding Logic components. The Logic component interprets the user input and sends it along to the corresponding core component Logic class. The purpose of having this class as opposed to direct connections between core component GUIS and their Logics is that sometimes input needs to be used by two separate Logic components. For example, if a grader enters a grade in the rubric, that must be carried out in the Rubric Logic and the Grade Book Logic.

Core Component Logics

Grade Book Logic: The Grade Book Logic carries out the functionality of the Grade Book including:

- Add/Change grades for an assignment.
- Add/Modify Assignment Columns
- Calculate grades from other grades
- Sort Grades
- Calculate Basic Statistics about Grades

This functionality is fairly simple, as it simply requires communication with the database.

Rubric Manager Logic: The Rubric Manager Logic carries out the functionality of the Rubric Manager, including:

- File I/O: Rubrics must be saved in a format recognized only by Teaching Tools.
- Handle grader input by communicating with the Database. Each assignment rubric will have a table in the database containing all information previously entered by a grader.

Notes Logic: The Notes Logic carries out the functionality of the Notes component, including:

- Storing current postings in database and handling new posts by users.
- Maintaining e-mail lists created by user. E-mail lists are also stored in the database.
- Must be able to use the default email program on the computer to send out emails using stored email lists.

Student & TA Information Logic: The Student & TA Information Logic carries out the functionality of the S&T Information component, including:

- Communication with the database, for retrieving, adding and editing data.
- Sorting capabilities of displayed information.

Document Creation Logic: The Document Creation Logic carries out the functionality of the Document Creation component, including:

- File I/O: created documents must be saved in a format generally recognizable, such as pdf format.
- Retrieval of templates for use in document creation.

Data

Database: The database is the heart of the program, because the program is very data-centric. There are tables for each component of the program except Document Creation. The database stores all data it receives from the Logic components.

Templates: The templates are used in document and rubric creation. They specify stationary style, text style and input style.

EXTERNAL DEPENDENCIES

There are two mild external dependencies – the email capability in the notes section and saving documents in pdf format.

The Notes Logic must be able to communicate with the computer's default email program simply to pull up an email window and place the appropriate email list as the recipient.

TASK BREAKDOWN AND GROUP ORGANIZATION

Project Manager

The project manager is responsible for the organization and progress of the project. S/He will schedule meetings and keep in constant communication with team members about their individual progress. S/He should keep all group members updated on the status of the project. The project manager should know the status of the project at any point in time. If a group member is struggling, it is the project manager's responsibility to either assist them or delegate the work to other group members. The project manager is the final word in any conflicts or discussions. As such, s/he is responsible for making and meeting deadlines. (# *people = 1*)

User Interface Architect

The user interface architect is responsible for the creation and implementation of the user interface. A focus of this program is its user-friendly nature. The user interface is crucial in carrying out this goal. Before actual coding, the user interface should interact with potential users to test out designs and gain feedback. The UI architect is also responsible for communicating with coders about interfacing with the Logic. Communication between GUIs and Logic is constant; therefore, interfaces must be nailed down early. (# *people = 2*)

Coders

The coders are responsible for carrying out the functionality of the program, which is encompassed in the Logic components. They are responsible for creating dummy data for testing. Coders are expected to completely carry out the implementation of his/her portion of the project. This includes detecting and fixing bugs in the coding, component testing and system testing processes. (# *people = 4*)

Documenter

The documenter is responsible for all project-related paperwork, including user documentation, testing documentation, bug reports, user feedback and project progress reports. Since the program is supposed to be extremely user friendly, user documentation should be very thorough. All functionality should be well documented with step-by-step instructions. There should also be sample written tutorials and exercises to help the user become better acquainted with the program. (# *people = 1*)

Tester

There are two testers: one responsible for technical testing and one responsible for user testing. The technical tester should run stress tests on the coders' work throughout the coding process. They are responsible for running the integration testing and system testing, including communication with the documenter. All testing should be well documented. The user tester is responsible for communicating with potential users and presenting feedback to the group. Initially, the user tester will research user's ideas about

the project idea and design. After implementation is complete, the user tester will sit down with potential users and help them use the program. All user feedback, including complaints, should be well documented and presented for the group. After assessing feedback, the user tester should brainstorm potential improvements and add-ons for the project. The user tester will also assist the documenter in writing user manuals and documenting feedback. (*# people = 2*)

ASSUMPTIONS

No assumptions have been made or introduced since the specification stage. As the project has not changed in functionality or design since the specifications, nothing needs to be clarified.

SCHEDULE

2/28 – Top Level Design Selected

- Project Manager selected
- Project Manager assigns group members to tasks and components

3/9 – Final Top-Level Design

- Design is finalized and cannot be changed without Project Manager's consent
- Task breakdown is finalized

3/12 – Interface Proposals

- Coders and User Interface Architects propose a model for interfaces between components

3/17 – Interface Comments

- Group has read and commented about potential interfaces. Project Manager makes thorough comments and meets with coders and UI architects to discuss interface proposals. Documenter keeps track of comments and progress.

3/9 – Presentation on User Surveys by User Tester

- User presents a survey for potential users to the group. The group gives feedback for information that should be included or discarded.
- After this point, the user tester should begin interviewing and surveying users formally.

3/21 – Final Interface Definition

- Project manager finalizes interfaces, ensuring that they are feasible and accurate.
- Interfaces cannot be changed without careful consideration and approval from the Project Manager.
- Documenter keeps written commentary about how the group's feedback was addressed and the changes made to the initial interface proposals.

4/4 – Detailed Designs

- Coders and UI architects should have a detailed design of their respective components. Interaction between components should be finalized.
- Project Manager should meet with coding group and approve design plans.
- Documenter should keep a written log of meetings, design concerns and the designs themselves.

4/9 – User Tester Presentation
- User tester presents feedback from potential users to the group. Feedback is addressed.
- Any change in program functionality resulting from user feedback must be carefully reviewed and approved by the project manager.

4/16 – Initial System Integration
- All components should integrate without full functionality. For example the GUI for each core component should be functional and communicating to the corresponding logic component through the main logic.
- Group meets and provides feedback and comments about the initial viewing of the project.
- Documenter keeps written representation of problems and concerns with integration process.
- Testers begin testing basic functionality as it is completed. Records of bugs are kept and addressed on a daily basis.

4/27 – Full System Integration
- Basic specifications have been met and implemented. Only small bugs should remain.

4/28 – Technical Tester Begins Thorough System Tests
- Technical tester runs stress tests and attempts to locate and repair bugs in preparation for public demo.

5/2 – Public Demo
- Public project demo in Lubrano.

5/3 – Assess Public Demo
- Any problems or bugs encountered in the public demo should be identified and corrected.
- Documentation should be complete and polished.

5/9 – System Submission
- Final submission of project – all coding, testing and documentation should be completed.
- Party!

VISUALS (from Specifications)

Diagram A: Graphical User Interface

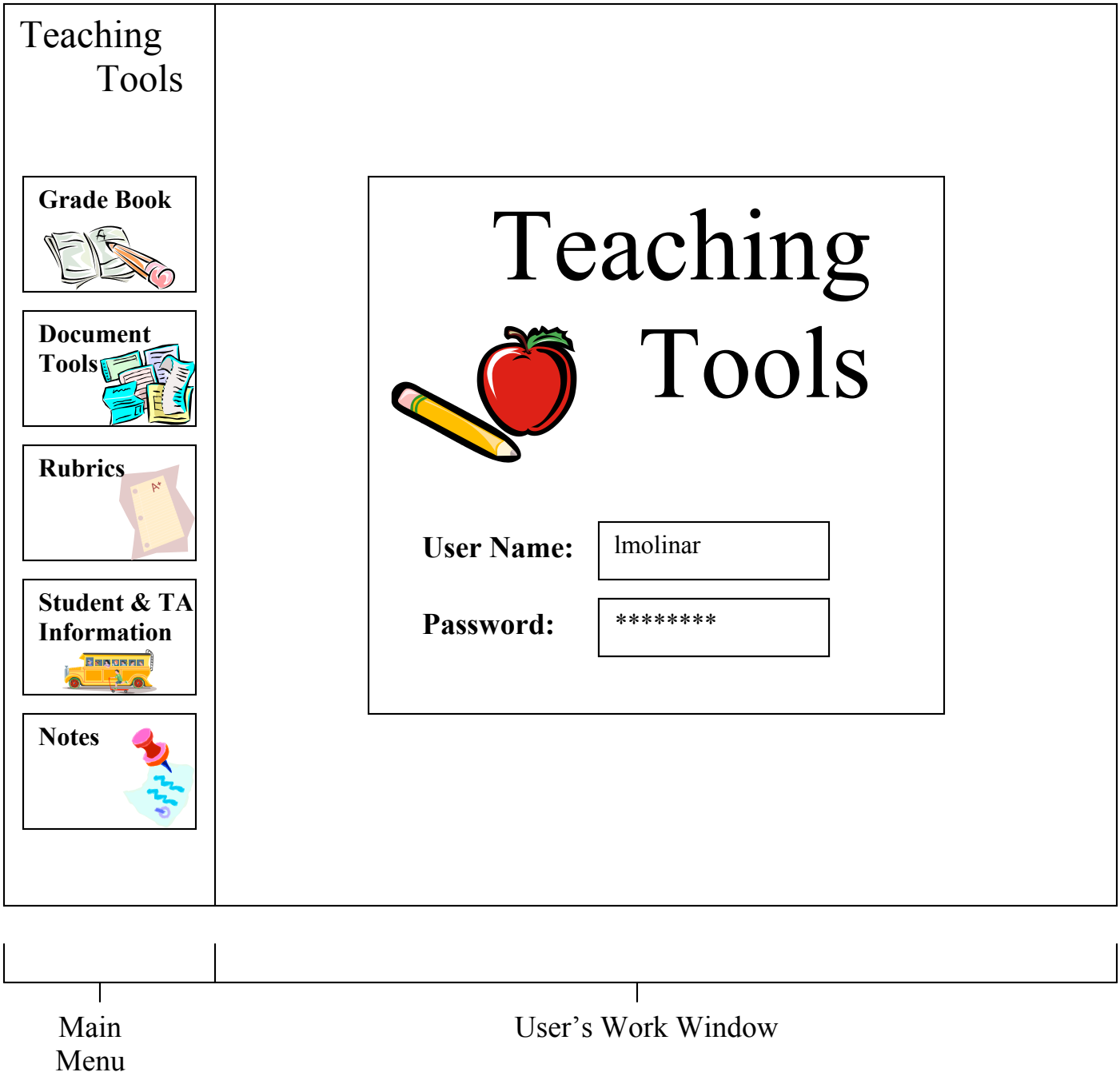


Diagram B: Grade Book Work Window

Grade Book

Modify Grade Book

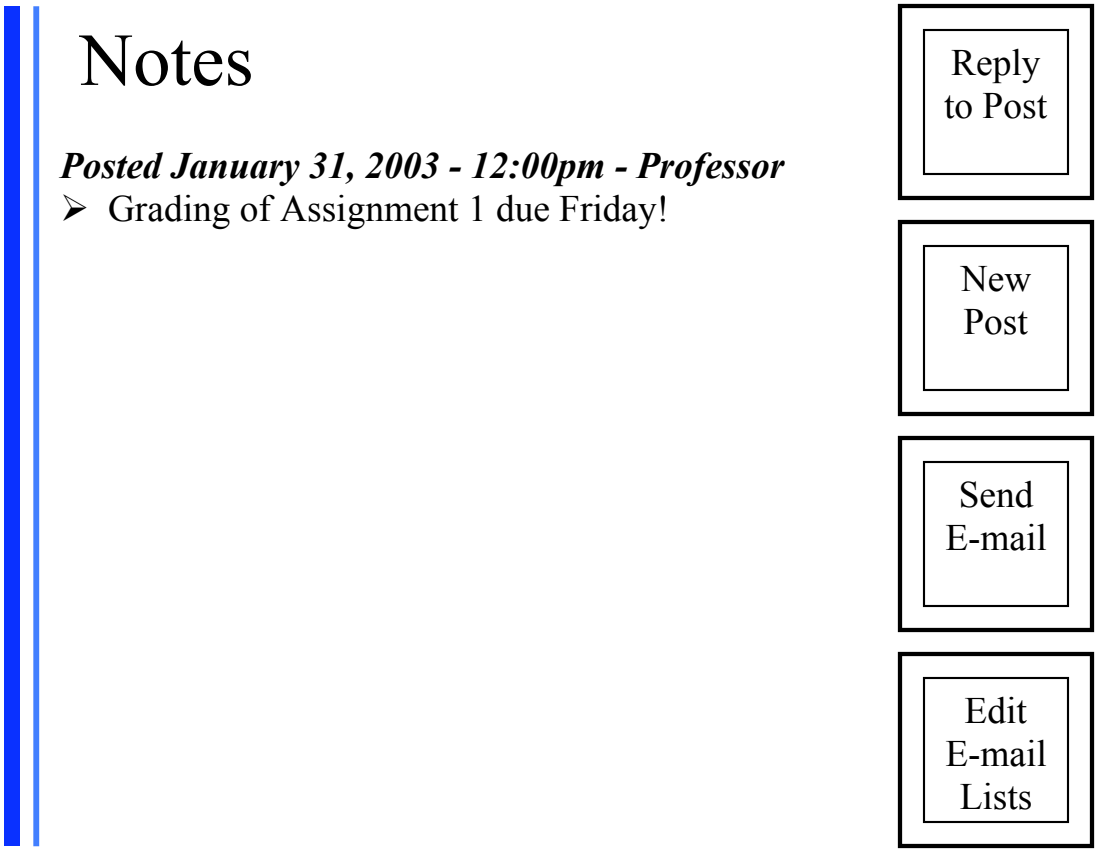
Add/Delete Students

Sort: A→Z Z→A

Modify Grades for: Assignment 1

Last Name	*First Name*	*Assignment 1 (/ 50)*	*Midterm (/ 100)*
Anderson	Billy	42	88
Barkley	Charlotte	48	95
Clark	Samantha	50	99
Donaldson	David	41	82
Farmer	Susan	47	92
Hahn	Alexandra	39	77
Hope	Haley	36	74
Johnson	Bob	49	90
Parker	Charley	50	93
Richardson	Kristin	47	87
Smith	Deborah	43	85

Diagram C: Notes Work Window



Sample of E-mail List Creation Window:

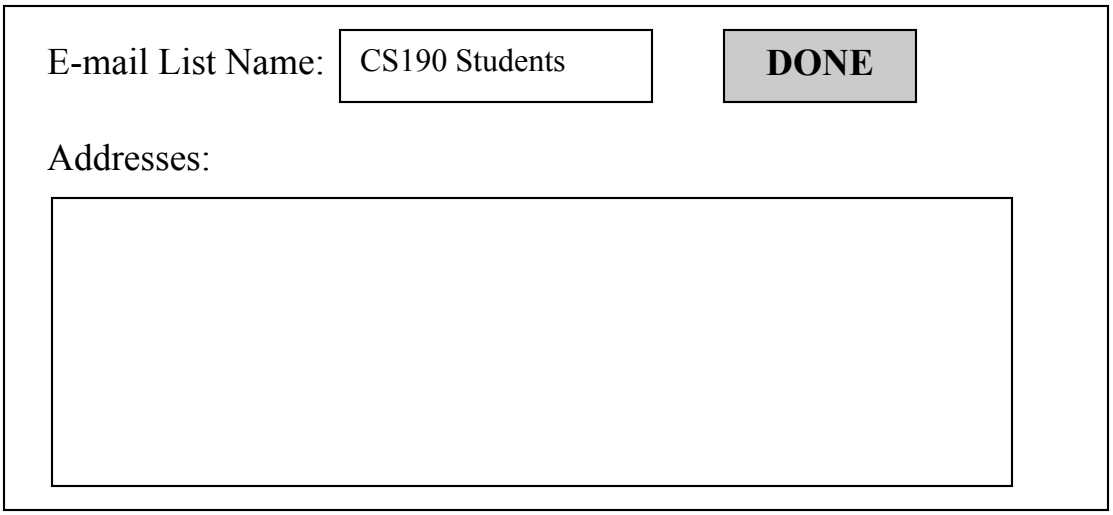


Diagram D: Example Templates