MRD Design Studios' Gradebook

Dennis Warner
Northeastern University
warner.de@husky.neu.edu

Mitch Drew
Northeastern University
mitch.drew@googlemail.com

Rich Van Buren
Northeastern University
vanburen.rich@husky.neu.edu

Abstract: this paper is written to outline the decisions and evaluations that went into the design of the "Whats My grade?" interface published by us for IS4300. It will make more clear the steps to our process and the refinements made along the way.

Author Keywords: When we make a reference to heuristic evaluations we mean an activity done as a homework assignment where our classmates were asked to apply Nielson's usability heuristics to the iteration we had published at that time. The heuristics can be reviewed here http://www.nngroup.com/articles/ten-usability-heuristics/

The paper prototype was another homework assignment where we were asked to mock up a version of our interface to be evaluated by other class members in class.

Problem: The problem that we are trying to solve is that students at all levels that want to keep track of the grades that they receive to chart progress or plan for the future. The alternatives available to students are a custom spreadsheet or Blackboard(Northeastern specific). The custom spreadsheets can take a while to construct and the special equations can prove tricky to unfamiliar users. The Blackboard system is difficult to navigate and not universally used by professors. Our proposed solution would place the responsibility on the students to make sure they are following the steps to achieve the goals they set.

Users:The users of our system come from all ages, grade levels, majors, and some non students. Our primary users are students but different types warrant different implementations. The first user is a student who is not as concerned about their grade either a good or bad student but still needs to maintain a certain grade to pass or stay consistent. These students would want to check their grades intermittently throughout the semester at a maximum or three or four times usually right before the midterm or final. These users will only be interested in the final output of the system and will perform differently on important assignments based on the results.

The other type of student that would use a solution like ours is one who is obsessive about grades. These users will be the type who are adding to their profile every time the teacher or professor returns a grade to make sure they are making the necessary efforts to get the grade they want. They will use the solution upwards of ten to fifteen times over the semester depending on how many assignments the course has.

The third classification of users are secondary like an overbearing parent or an academic adviser. The parents will fit more closely with the users who are constantly updating class profiles to make sure their child is doing their work. Hopefully this would not be parents of college students but it remains an option in that case. The academic adviser would fit more with the student who only looks a few times a semester but would only really have to use it one time. If a student came to them with concerns about their grade they could quickly calculate the students grade and provide advice according to the results.

Tasks:Independent of the users there are certain tasks they all need to accurately represent their performance in a graded class. The process starts at the beginning of each semester by adding classes to the users profile. This is the starting point for all other tasks and users should be able to specify name of the class and other optional information associated with that specific class. If a mistake is made in the adding process there should be an easy way to go back and edit these fields. At the end of the semester users will likely want to make room for new classes so they should be able to easily delete classes and all information associated with it. These tasks go together and will be done once or twice a semester depending on the users course load or if they add or drop classes.

Once a class has been created, a user needs to add fields that they are graded in in the class. This information is usually provided by the syllabus and would include things like: tests, homework, team projects, classwork, etc. The syllabus will often also contain weights that are associated with each of these distinct grading areas and will be used for calculating the final grade. These areas can be deleted or edited by the user if they made a mistake or requirements change. The frequency of these tasks depends heavily on the number of classes and the types of classes that the user takes but on average three to four times per class.

The main task for the users will be adding and deleting grades. Mistakes when entering grades is undone by simply deleting and reentering the correct value. The frequency of this task is dependent on a number of factors including class type, subject, and professor diligence. It could be done anywhere between five or fewer to twenty plus times per semester.

Design: The final design of our interface takes into account a lot of feedback from members of our class as well as design decisions made as a group. The final interface is not fundamentally different from our first iteration because of the extensive considerations made at the outset of the project. We considered a few alternate designs(fig 1 and 2) that each would have served the purpose but not to the extent we hoped for.

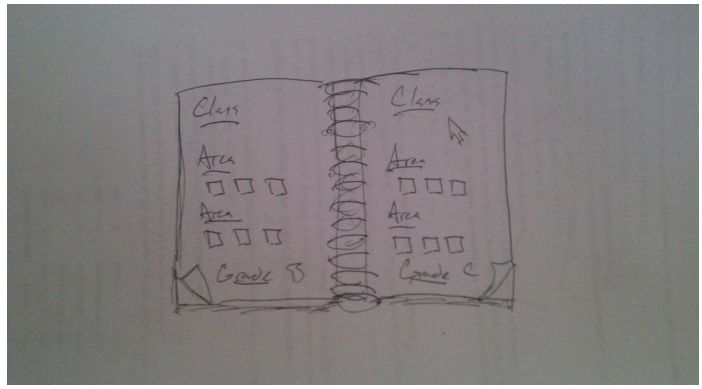

Figure 1. The emphasis of the grade book methaphor

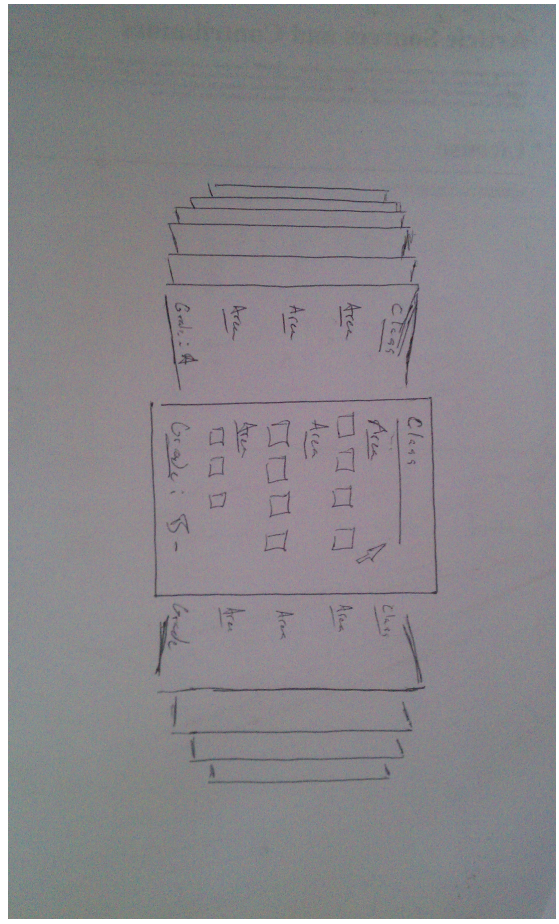

Figure 2. The scrolling based interface

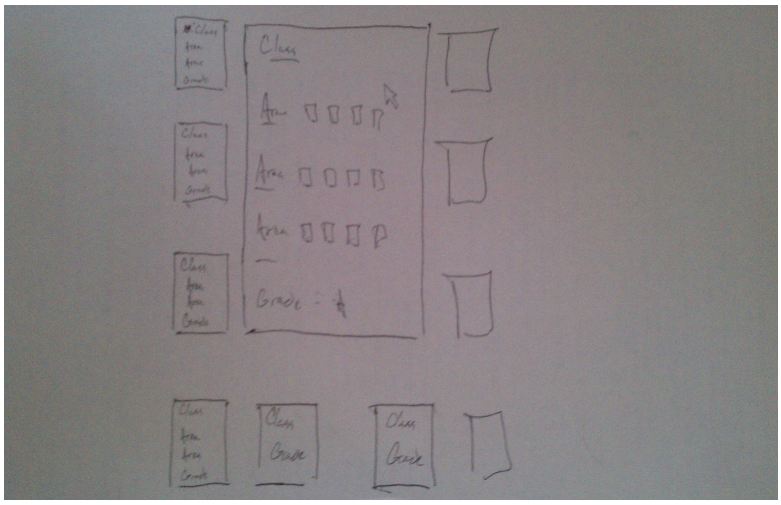

Figure 3. The interface we ultimately decided to design around

The first fits with the grade book metaphor and is pretty aesthetically easy to understand at a glance. But only showing two classes at a time limits the visibility and there is chance you have to search through all the pages to find a certain class. It posed unique problems as well like determining if people would know the up-turned pages would turn them. We also struggled with finding an efficient way to add new classes.

The second design improves on some of the problems of the previous one but does not completely solve them. You can barely see all the classes at the same time, but it is more consistent with familiar interfaces of various iOS devices. The problem of adding classes still exists and we did not want to add buttons outside of the scrolling panes.

We ultimately chose the design pictured in figure 3. This solved all the other problems the other two had. All classes are visible at the same time and adding classes is done by selecting an empty pane. The two alternatives would be infinity expandable by adding more pages but we decided that is not necessary because the typical student, high school or college, will rarely need more than eight classes at a time. The classes along the edges are clickable to bring them into the foreground where the add, drop, and edit options for the specific class are done. Our first iteration was fraught with errors that was improved with the help of our classmates.

Paper Prototype: The paper prototype was a more complete version of our original design with a few user experience problems that were pointed out after one run through of the tasks and confirmed by the second group of test users. Right away we had two major problems with navigation. The original version displayed a welcome message with pretty much no information on it. The users did not know how to start the process so they had to ask. This was a severe bug because if they could not get that far, there was no hope for the other tasks. This was solved by putting a message on the welcome page that said “Select a class or start a new one” just to give them some idea that the data that was already populated was just for the testing purposes. The second most common comment had to do with the fact that one of the tasks asked the users to go back into a class to edit the fields and add a few grades. They were able to get to the class and change things accordingly but then got stuck. Originally the idea was for it to update live as changes were made which was an oversight on our part and so we clearly had to add some sort of “save” or “commit” button (figure 4)to the class page to signal that they have completed those actions. After hearing this from many users we added it to each class page and when it was clicked it would return to the home screen and users could see in the small thumbnail window if their changes had been made. This made us give more thought other interesting considerations such as having a way to return to the home page without saving the changes but we decided to leave that out in favor of just having them move to another class. Once these major factors had been taken care of we mocked up a computer version for more testing

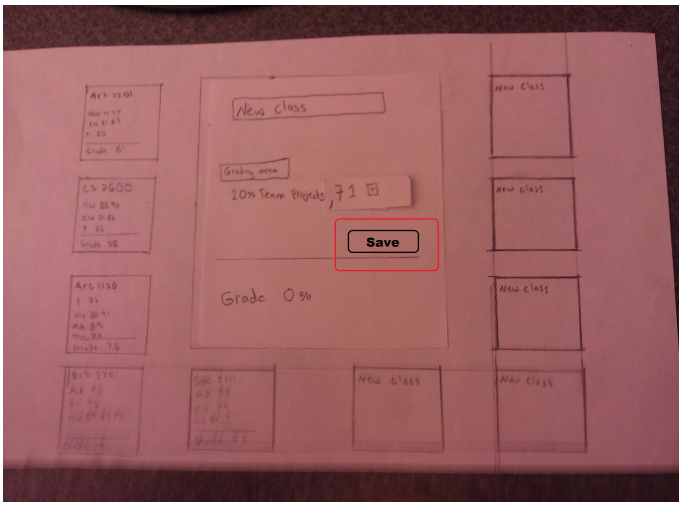

Figure 4. The proposed location for the save button for each class

Computer Prototype (Heuristic Evaluation): The major concerns had been addressed at this point so the heuristic evaluations would be able to help us iron out the smaller usability errors. A lot of concerns centered around the wording of labels for the various areas. We wanted it to be as simple as possible so displayed instructions were kept to a minimum so the short sentences that served as explanations had to be compact and we oversimplified and made assumptions that our users did not. As a result we had a number of things to rewrite including the grading fields and a special instruction in the class page. When the user originally created a new class they would have to submit at least one grading field and so in the first implementation there was an input area labeled "field 1". This was misleading because there was no field 2 and users did not know what this meant at all. We changed it to say "Grade Area" and in the input put a placeholder of "Homework, tests, project etc". This was done in hopes of steering the user to understand that these are the areas you are being graded in not an area for individual grades. The other comments were about some other basic phrasing but a few had to do with the deletion of classes from the home screen. Their reasoning was that classes are created from the high level so there should be a way to delete from the welcome screen. We experimented with this idea and added an "x" at the top corner of the thumbnails to signal their deletion. This did not really fit with our idea of modifying the classes from within them and we did not want to have a dialog box pop up and cover the screen to confirm the delete. Instead we added the delete button in each class page and the dialog appears at the bottom of the class window to confirm the deletion (figure 5). This seemed more consistent with other methods for modifying data about the classes.

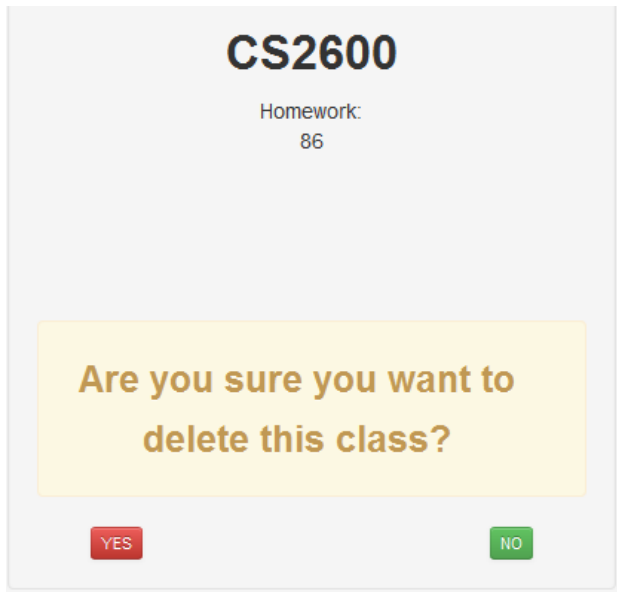

Figure 5. Solution to dialog covering the class page

High Level Considerations: The core factors that we designed around were that the final product had to look visually simple while remaining scalable for custom usage. There are really only two "pages" one being the home screen and one for when the user is modifying a class. These are easily transitioned between one another and the different pages for the different classes. There are very few instructions and it is designed so that the user can see what is happening at all times. The thumbnails around the main frame keep track of changes to other classes so the users will know what state they are in and modifying it is intuitive. From both forms of evaluation we realized that the minimum instructions means the phrases use have to be concise and steer the user to the right

area to figure out the next step. The simplicity also forced us to encode some of our assumptions that turned out not to be universal leading to redesign in the later stages based on classmate's inputs. The major design barriers seemed to come from the prototyping tool we used and as a result we couldn't make certain features in a way that adhered to our basic principles.

The first and most important design decision presented itself right away and would steer the design of the project from then on. We had to decide which user group we wanted to target and build the interface sot that their tasks could be easily accomplished. The two groups boiled down to those who would use it a few times and only want the result or the people that would consistently update their profile to ensure they were on track. The first group would want the ability to load a lot of grades in at one time and quickly add classes for a one time use. They would then use just the information that the interface told them was the final grade. This was deemed the calculator approach because it only for a few quick calculations and then would be discarded. The second approach was the users who would add grades as they got them back and create a plan to budget their time based on the results. These users would only need to add one or two grades at a time and add or delete classes once per semester. We decided to design for these users as opposed to the calculator approach. The persistent users would benefit much more from way to catalog their grades and shoulder the responsibility of making sure they were on task. The group that only wanted the final grade could easily just use a weighted average equation to get what they needed.

Implementation: Implementation of our project at a high level was very simple but the particulars of the code increased the complexity. After we had taken the steps outlined in the Design section, we knew we would have to create a web site and we wanted to use HTML and CSS. The comlexity ramped up faster than our introductory knowledge so we were forced to use a prototyping tool to make sure we were able to finish on time.

After we were done designing our interface, we began to create the front end using HTML And CSS. We were using a program that uses Twitter Bootstrap and an interface to create the web page “WhatsMyGrade.info”. The program was called Jetstrap, and I was working very easily as it took out a lot of the complexity of creating websites using HTML and CSS. One problem was that we faced while creating our program was that none of us had much experience in web development. So everything we made took us a lot of time and effort learning to create. Even with the web interface of Jetstrap, creating the website was difficult.

When we were building the web interface for our program, we tested our design multiple times and received different feedback each time. And each time we received feedback about something that should be fixed, we would take an objective approach to suggestions and make sure that they stayed within the high level goals we set at the beginning. This was sometimes very complicated as we implemented and tested many different variations and compared them against one another with the help of classmates and each other. On a few occasions it came down to a tradeoff between user experience and the simplicity that we wanted to convey. This was often times difficult as we knew what we wanted but could not figure out how to create what we wanted to make.

One setback with Jetstrap was that in the beginning it was a very buggy web interface crashed multiple times before it was made stable. Another set back with Jetstrap was about two thirds through this project in which they monetized their project so one had to pay to use their CSS feature which caused us to lose all of the CSS we had already created for the site and forced us to recreate the CSS to make the site look

good and feel usable again.

Evaluation: We conducted our test in multiple ways. In the beginning we used paper prototype, then we switched to very basic user testing just to see what would happen when we asked someone to complete a task. Finally we ask someone to complete the tasks and watched to see what they did that we did not expect and figured out how to solve that problem.

The three sample tasks we gave:

1. Create a new class called IS4300 where homework counts for 10%
2. Open CS2600 and add a 71 to the homework section
3. Re-open IS4300 and add team projects worth 20% and add an 83 to the homework section

These tasks were designed to have the users change the state multiple times and be able to apply the same basic steps but in a different setting. In addition they would serve to validate some of the design decisions we made previously.

When we were using a paper prototype, We were still in the process of completely exploring all alternatives and it was very helpful to ask different people to complete the same tasks but with an interface that we had the ability to constantly change. This allowed us to better understand what people though and what made the most sense to the largest amount of people. Figure 6 shows a user completing the tasks with the paper prototype

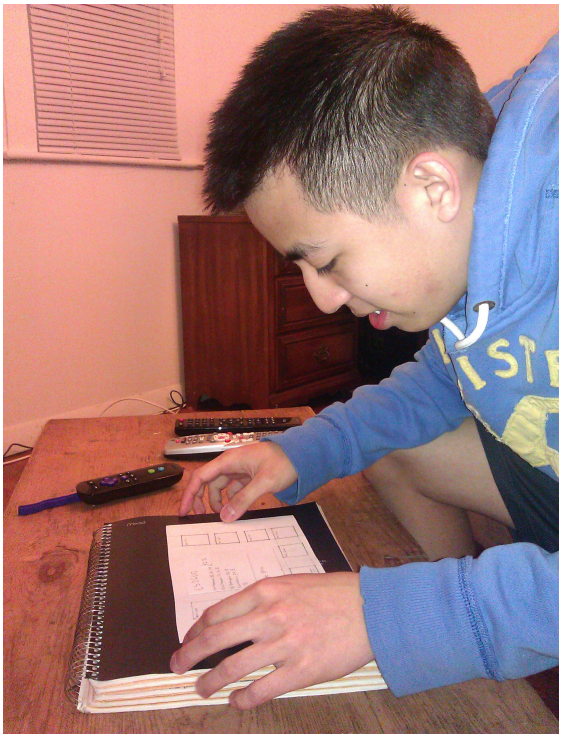

Figure 6. User interacting with the first iteration of our paper prototype

The process broke down to one group member would give the tasks on note cards that had been prepared prior to the start of class that had a sample task from above. That same person was in charge of moving the pieces around as the test subjects interacted with various parts of the system. On other group member was in charge of watching the user specifically to focus on the body language looking for things like prolonged stages of confusion or frustration. These were then accounted by the last member who was in charge of recording the results from the observer. They also took notes themselves about a few more specific interactions like misclicks or questions that they asked throughout.

After we had the large details and most of the big details hammered out, we set to work building our interface. As we were building our interface, we would ask people to test out the interface by doing simple tasks in addition to some basic visual schemas. Some simple tasks were just adding a

class, or adding a grade or removing a class. And We would ask people to do one of these simple tasks, stand back and just watch to see what they did. These were much more informal and more for quick fixes. We would answer any questions that they might ask but ask for their input on a solution as well. We would take what we saw and use that data to make the web interface easier and cleaner to use. We did this multiple times until we had a good usable interface where we could start asking people to doing true user testing.

In the true user testing, we created a prepared statement explaining the program and what it was for, what to do in the user testing (the tasks), and telling people that they were under no obligation to continue the testing if they chose to leave. And from this we ironed out a lot of small details to make the site look good, and be as simple as possible to use. We would have people sit at a desk or table, anyplace where they could use a computer or laptop, and to do the three main tasks. We would observe and offer no comment until the study was done. When they would finish we would ask some simple questions such as what was the easiest task, what was the hardest task and what do they think made it difficult. And only then would we answer questions about the program if no other participants were close enough to hear us.

Reflection:

The iterative design process allows implementers, designers and interface engineers to meet with their users on a much more frequent basis than a stereotypical waterfall design process. This truly opens up the interface design to accommodate a more substantive relationship to the end user, allowing for consistent and constant feedback which can be used to further tune the interface to meet previously existing or newly discovered design goals.

When properly applied, with user testing thoroughly included in all steps, iterative design will consistently result in a much more streamlined and efficient design than other processes, especially when measured against the waterfall process. At the beginning of our design process, we were focused on developing a slew of interfaces to test against each other. We focused on using our own judgment to figure out which would interface would be the branch that we develop into our final design. Using the iterative design process to its full effect, we would have interviewed and canvassed our original proposed user base for possible design goals, before we had even put anything to paper. This is certainly our first mistake, to not include the iterative design process in our interface design from the start. It led to us not using the process until much later when we felt that we had an interface that could be improved.

Had we realized that we could have fine tuned any and every part of our project with the iterative design process, I feel we could have avoided many of the more troublesome pitfalls we managed to step into as the process unfolded. We only managed to include user feedback into the loop once we had rolled out what we felt was a usable interface, instead of conferring with users and focus groups at multiple times previous to that. While it would be certainly difficult to call it a debilitating decision on our part, or to think that we would've ended up with a completely different result, it is certain that we would have avoided many of the more time consuming debacles that were associated with our interface design.

The interface also reveals the effectiveness of a minimal design. Looking back, our previous designs featured alluring visual details that could only bring the flow of interaction to a halt. As we further refined our design and came to our final idea of what it should be, we had removed the social module, the graphics, the login function and even the menus. As we stripped away from our prototype we found that our interface was left with only what it needed to accomplish the tasks that it was originally intended for and those only it

could do.

We as a team felt that the need for some of these modules was sparse, and their existence only added undue complexity for all users and tasks. Once we stripped our interface to a simple, totally visible system we found that there was little explanation that needed to be given. Users did not have to be given instructions on how to use the interface in an intuitive way, which significantly improves the usability of an interface. Iterative design certainly pointed us in this direction, by testing out small changes and removing small amounts at a time we were able to significantly change the nature of interaction our users had with our interface from one of mildly arcane to intuitive and simple, needing no directions.

Once we narrowed the scope of our user base and target tasks, our interface's utility sharpened accordingly. Our early prototypes for an interface whose target users included instructors and teachers was broad in scope and utility, but it became a system that was very difficult to parse at first glance. It necessitated a login feature that requires additional back end, and once we broke the two user groups into their two interfaces, we realized that they were two different projects altogether, with their own unique goals, users and tasks.

Evaluating and reflecting on our technique, I would say that our analytical and aggressive approach to rapid prototyping was successful and produced a very usable interface for a utile system. Our small team of test users quickly allowed us to evaluate the results of our latest prototype, letting us know immediately what worked and what didn't. While larger sample sizes and different test groups would've helped, the toolset we chose was more than effective at meeting out a satisfactory result.