2011

Designing for the User: Exploring the Interface Design of Web Services

Katelyn Anne Haw
University of Northern Iowa

DESIGNING FOR THE USER:

EXPLORING THE INTERFACE DESIGN OF WEB SERVICES

A Thesis Submitted

in Partial Fulfillment

of the Requirements for the Designation

University Honors with Distinction

Katelyn Anne Haw

University of Northern Iowa

May 2011

This Study by: Katelyn Anne Haw

Entitled: DESIGNING FOR THE USER: EXPLORING THE INTERFACE DESIGN OF WEB SERVICES

has been approved as meeting the thesis or project requirement for the Designation

University Honors with Distinction

4/29/11
Date

J. Ben Schafer, Honors Thesis/Project Advisor

5/6/11
Date

Jessica Moon, Director, University Honors Program

Acknowledgements

I would like to express gratitude to those who helped me throughout the development of my thesis project. Without their guidance, support, and encouragement, I could not have completed this thesis. I would especially like to thank my thesis advisor, J. Ben Schafer, for his hard work in guiding me throughout my process. In addition I would like to thank Jessica Moon, Michele Peck, Philip East, Stephen Hughes, Amye McInnerney, Jessica Steen, Laura Cross, Jessa Schweitzer, Leandro Avila, Michelle Cross, Kevin Shannon, Shirlee Haw, and Kyle Lensch for providing me with the necessary assistance and inspiration needed to complete my thesis.

Table of Contents

Introduction

The advancement in technology has created a broader use of electronic and computerized systems. Where one used to have the option to do something by paper, society is compelling individuals to complete tasks on a computerized system or on the internet. In the doctor's office, the bank, or even at one's workplace, a person is forced to use computer software and other technology to accomplish tasks. These tasks are often time sensitive and sensitive to large errors whether they are created by the human user or the program. For example, tax software that allowed the user to easily make large mistakes could have serious consequences financially and legally, thus increasing the need for a strong user interface that is easy-to-use and prevents errors. More and more people are using electronic means to complete common tasks. In fact, the IRS estimates that 122.3 million tax returns will be filed electronically by 2010 (Hussain, 2005). In addition, the number of mobile phone users accessing the mobile internet increased 73% from 2006 – 2008 (Roberts, 2010). The increase in technology has increased the demand for well-produced user interfaces. When one is forced to use a piece of technology that cannot be easily understood or used, one becomes frustrated. Users of such systems often blame themselves for errors when in fact the error was the fault of the system, not the user.

User Interface(UI) Design is the process of creating the part of software that interacts with users so that it is not only visually appealing but easy-to-use and intuitive for all users. In recent years, developing easy-to-use software has become more and more of a priority for software developers. Early on in the development of computer software, the ability of designers to make software easy-to-use was less important because the limitations of the hardware restricted what a UI could do. Now with computer hardware such as touch screen interfaces, multi-core processors, and high resolution screens, the possibilities for designing a UI have grown exponentially within the last decade.

With more possibilities for graphics, movement, and interaction between the interface and the user, UI design is becoming more and more of an interdisciplinary process. Software engineers often delegate the design of the interface to a group of graphic designers who will give the interface design back to the software developers to implement into software. With the complexity of graphic designers and software engineers working together on the software, it can be a challenge to ensure the goals and requirements of the system are fully communicated. Usability is more than merely how "pretty" an interface looks. The software itself has to be usable and it must have the functionality that correctly fits what the user needs. A piece of software with a visually appealing interface is worthless to a user if it does not function in the way expected, if it produces confusing errors, or does not do what the user thinks it should. To ensure that a system is usable, the user's needs should be considered at every stage of the development process. Without a well-developed design process that is focused on the user's needs and good communication between the different people developing a system, it is not likely to reach its full potential.

My combined background in both graphic design and computer science will give me a unique view on the user interface design process. Through my thesis project I gained hands on experience with all stages in the software development cycle including the graphic design process and the design and implementation of the software backend. The exploration of the interface design process led me to discover places where the communication between the designers and implementers may break down or where the consideration for the user's needs may fall short. With this valuable experience, I have gained knowledge to ensure that the functionality, performance, and usability of a system are all being fully considered throughout the design process.

Problem to be Addressed

Students are faced with overwhelming decisions when they begin their study at the University of Northern Iowa (UNI). Scheduling for classes can be one of the most daunting tasks students have to complete every semester. A lot of variables go into creating each semester's schedule of classes. Students have to determine what courses they have to take next in order to complete their majors, minors, and liberal arts core, all the while paying attention to prerequisites, course rotations, and university requirements. Programs can have very strict prerequisite requirements or courses that are only offered in certain semesters, putting more pressure on students to make the best decisions when scheduling. A decision made about one semester can have a substantial effect on all following semesters.

UNI has an undergraduate plan of study application that is available to students through their myUNIverse accounts. Although the current system provides students with a basic tool for layout of courses needed for their majors, it is not customizable or flexible enough to meet students' needs. Much improvement is needed to make the current system better fit the needs of students. In order to explore the usability design processes, I created a web interface for a Plan of Study Application. This problem offered many interesting possibilities for developing an interface. There were many usability, functionality, and organizational questions that needed to be address through my design processes.

Purpose

My thesis investigated the ideas of usability and the design process to create software that is considered usable. I developed a web interface for a Plan of Study Application that would help students make the educated decisions about their studies at UNI. Throughout the development of this application I addressed the following questions: *"What makes an interface usable?"* and *"How does the design process affect a system's usability?"* User interface design

is an interdisciplinary process and it often involves computer programmers and graphic designers working together. Without clear communication and vision between the groups, the software being developed has little chance of reaching its full usability potential. Throughout my thesis project I discovered how graphic designers and programmers view software development differently and how the communication between them might be improved. I reflected upon how my unique education as both a graphic design and computer science major offers me a distinctive insight into the design process.

Source Review

Defining Usability

Experts in the field of user interface design have their own opinion on exactly what makes an interface usable and what defines usability, but their definitions of usability have one thing in common: the user. According to Garrett (2003), usability is about helping a person complete a task faster and guiding them to make fewer mistakes. He stated that technology that does not work the way a user expects can make him or her feel stupid or incompetent even if the technology actually accomplishes what it is supposed to do.

One of the big mistakes often made by UI designers is to assume that a user can adapt to anything (Johnson, 2008). Psychologist Donald Normans believed that a user should not have to adapt to a poor UI. He pointed out the irony in situations where a user has a problem using an interface and instead of blaming the devices' poor design they blame themselves for the error and become ashamed to have triggered it. He asked the question "Why [do] people blame themselves when a device itself was at fault?" (Norman, 1988, p. ix). Kransberg said that "Technology is neither good nor bad; nor is it neutral," but without a good interface design, technology is more likely to be good than bad (as cited in Buxton, 2007, p.38). Ultimately designing something that is usable is not merely just designing an interface, it is designing the

experience the user will have with the system (Buxton, 2007). An important aspect of that experience is how easily the user can learn the interface, but Johnson(2008) points out that usable means more than just being easy to learn. He makes the claim that being usable means that the software is easy to use, quick to use, relatively error-free, and most importantly that it does what the user wants it to do (Johnson, 2008).

Aspects of a Usable Interface

The usability of a system is dependent upon many decisions that a UI designer must make. Norman (1988) described the aspects of a usable interface in a series of design principles. He stated that a system should have good visibility, a well-developed conceptual model, good mappings, sufficient user feedback, and appropriate constraints. Good visibility means that the user can determine the state of the system. A system with appropriate feedback will inform the user that their actions have done something in the system, and good constraints will ensure that a user cannot make a mistake. Good mappings mean that there is a good visualization of how a user's actions cause an effect within the system (Norman, 1988). A conceptual model is the designer's model of how the system works and how they want to present the system to the user(Johnson, 2008). A mental model is how the user perceives the system to work (Norman, 1988). A user develops their mental model by drawing on their previous knowledge to understand a new interface (McCracken & Wolfe, 2004). In a system with high usability, the designers' conceptual model should be the same as, or very close to, the user's mental model (Norman, 1988). Designers often communicate their conceptual models of the system to their target audience as a metaphor. This allows users to relate an interface with something that is familiar (Eaton, 2003). Eaton pointed out that metaphors are particularly important in website interfaces because users only spend short amounts of time on very specific tasks, and metaphors can help users understand what they need to do faster (Eaton, 2003).

Johnson (2008) also defined a set of basic design principles for user interfaces that include: conforming to users' view of the task, designing for the common case, not complicating the user's task, facilitating learning, delivering information, designing for responsiveness, and trying it out on users. Conforming to the user's view of the task includes striving for a sense of naturalness in the controls of the system so that the user is not forced to do something that feels unnatural to them. It also means using the user's vocabulary and keeping the implementation of the system hidden from the user (Johnson, 2008). Designing for the common case is making the tasks that are done the most frequently or by most users the easiest tasks to accomplish within the system (Johnson, 2008).

User interface design is more than just an attractive look to a piece of software. Johnson states that an interface "embodies design decisions that extend down deep into the architecture"(Johnson, 2008, p.19). He argued that decisions made about the concepts of a system early in the design process can also affect the system's overall usability. According to him, an interface's usability is what concepts are exposed to the users, how the information is structured, the customizability of the system, and the backend functionality of the system. Johnson's idea is that usability and the needs of the user should be the focus of all stages of the design process (Johnson, 2008).

A Usability Design Process

Focusing on the user is the whole idea behind the task-centered design process created by Lewis and Rieman (1994). Lewis and Rieman (1994) created a design process they titled *Task-Centered User Interface Design*, which is focused on creating a system based on a set of tasks that a user should be able to accomplish with the system. Johnson (2008) also mentioned how a designer should consider the function first and the presentation later. Lewis and Rieman's

design process includes: a requirements analysis phase, a specifications stage, the initial designing, evaluating the system without users, and then user testing.

The requirements analysis should be defined in terms of the users and the environment and be comprised of real, complete and representative tasks that the user ought to be able to complete in the system (Lewis & Rieman, 1994). Two common design myths are that designers know what they want to build in the beginning of the process and that they know enough to start building right away (Buxton, 2007). The requirement analysis phase is the part of the design process in which designers discover the answers to what they will build and who they are building for. During this phase, designers complete a user analysis to discover their users' demographics, learning style, and tool preferences (McCracken, & Wolfe, 2004). A designer must also consider the users' general computer knowledge, knowledge of the tasks, and knowledge of the system (Johnson, 2008). Not all users of a system are created equal. There is an unequal distribution of the internet among poorer and richer communities, men and women, and minorities, which can lead to groups of users who may view the same system differently (Shneidermann, 2002). Shneidermann (2002) suggests that software can be designed for different groups of people by designing it in a multi-level system. By designing software in different levels, we can give each level a different set of functionality based on its intended user Each level in a multi-level system can be designed for a different user's skill level or a different ethnic group (Shneidermann, 2002).

The specification described in Lewis and Rieman's (1994) design process contains the details needed by software designers and implementers. It takes the tasks described in the requirements analysis and adds details about how they will be performed (Lewis & Rieman, 1994). Garrett (2003) stated that a functional specification should be positive, specific, and ought to avoid subjective language.

Lewis and Rieman described two different tests to be completed prior to testing with the system's users. The first test is a cognitive walkthrough of the system where a tester uses the tasks for the system defined in the requirements analysis and the initial interface design and they think through the tasks to identify usability risks. A cognitive walkthrough can be completed when the interface design is only in a wireframe state (Lewis & Rieman, 1994). The second type of evaluation without users is the heuristic analysis of the interface. During a heuristic evaluation multiple evaluators will test a prototype of the system and evaluate it according to a set of heuristics (Lewis & Rieman, 1994). There are nine heuristic principles that are generally recognized as the industry standard for measuring usability. These were developed by Molich and Nielsen (1990) and included: ensuring that the system uses simple and natural dialogue, speaks the user's language, minimizes the user's memory load, is consistent, provides feedback, provides clearly marked exits, provides shortcuts, provides good error messages, and has good error prevention (Molich & Nielsen, 1990). Each evaluator should comment on the interface independently and later collectively discuss findings as to not create a bias in their individual evaluations. Although heuristic evaluations only catch somewhere between 20 and 50 percent of usability errors, they are a cheap, intuitive evaluation that can be done early in the design process (Nielsen & Molich, 1990).

Iteration is essential to a task-centered design process (Lewis & Rieman, 1994). It allows for refinement of the interface to occur which can help keep the user's requirements central to the design (Bowen & Reeves, 2009). Iteration also will ensure that any user feedback received during testing will lead to amendments to the interface. Bowen and Reeves (2009) stated that refinement is a formal process which transforms one system into another. Refinement guarantees that the original properties of the system are preserved and each iteration decreases the level of abstraction within the system (Bowen & Reeves, 2009).

Norman (1988) described some of the factors that may inhibit a good design process for an interface. He stated that a lack of time, focusing on an immediate problem instead of the big picture, the idea of individuality, and creeping featurism all are forces against good design. Norman also warned that designers themselves may hold a system back because they can be too focused on aesthetics and they are not the primary users of the system (Norman, 1988).

Methodology

The implementation of my thesis project followed a modified version of the design process described by Task-Centered User Interface Design (Lewis & Rieman, 1994). During the requirements gathering phase, I researched the current system as well as both the needs of the system and its users. From this analysis, I created a *requirements analysis*. I took the requirements I discovered and created a *functional specification* which explained *"how"* the requirements would be met in the system. During the design phase, I designed the system's interface through a series of wireframes. I then implemented the interface. Throughout the design process, I completed a series of usability test which helped me to ensure that I was designing and building my system with the user in mind. I completed a *cognitive walkthrough* of the black and white wireframes, a *heuristic evaluation* of the initial prototype, and *user testing* of the final system prototype. Each test discovered areas of the interface and system that needed to be reworked and redesigned to improve its usability.

Problem Analysis

As I began the analysis of the problem, I was unaware that a system already existed to help students create a plan of study. My lack of knowledge that such a system existed led me to question why the system is not widely known about and how it could be improved. I did an

extensive analysis of the current system (See Appendix A: Analysis of Existing Undergraduate Plan of Study) in order to get a better sense of its strengths and weaknesses.

The University of Northern Iowa has had a Plan of Study application available to students since the Fall of 2003 (*Undergraduate Plan of Study*, 2008). The Application was developed as a way for students to create a plan for their studies at UNI. It was intended to make the scheduling process more efficient and help students think ahead to future semesters. The current system is still available to students through their MyUNIverse accounts.

The current system does not meet the needs of students. The application merely merges standard major worksheets to build a plan. It allows students to add courses to their schedule, but it does not allow students to move courses around within their schedule. The application is only useful for freshmen or sophomores who are trying to determine which major they are going to pursue because the application breaks if students have not completed the courses that were suggested in previous semesters. If a student misses courses in the plan, the courses are added to a holding area at the end of the plan. This makes the application completely useless for juniors, seniors, or transfer students interested in adding another major or switching their major. According to Academic Advisor Michele Peck (personal communication, January 2011), the current plan of study application is most useful for freshman or sophomores who are starting a major. She stated that the current system does not work well for juniors or seniors. From my analysis of the current system, I was able to conclude that there were multiple problems with the current system and much room for improvement, particularly with the system's interface.

Creating the Requirement Analysis & Specification

The user interface design process begins with an analysis of the problem, the users, and the tasks the system will need to be able to complete. A *requirements analysis* is meant to be a broad view of what the system should do for the users. A *function specification* breaks down

those tasks and explains *how* the system will allow the user to accomplish them. Most large projects to build computer systems use *requirements analysis's* and *function specifications* to describe what their system will do and how it will do it. In the Task Centered User Interface Design (TCUID) methodology, these documents are focused on the users and all of the requirements and specification for the system are worded in relationship to the user.

I began my analysis of the undergraduate plan of study application by conducting a series of interviews. I interviewed a freshman chemistry major and a senior social work major to determine how they currently go through the process of scheduling and what problems they encounter. Both students used paper and pencil methods to keep track of the courses they needed to take rather than using the current Plan of Study application. In fact, neither student knew that there was a current system available for their use. To keep track of the major courses she still needed to take, the senior used a flow chart diagram produced by her department and given to her at a major scheduling meeting. She kept track of her Liberal Arts Core (LAC) courses by checking them off of the listing in an old newsprint schedule book. The freshman used information about her major requirements she found online and a paper listing of the LAC courses that she received at orientation to help her determine which courses she should be scheduling for next. She also noted that a plan of study will often change if all sections of a course are full in particular semester or a student is unable to fit the course in for various reasons. This emphasized the need for the plan of study application to be flexible and allow students to continually modify their schedule. The limitations of the current system do not allow for a student to rearrange their plan of study and it will not handle courses that could not be fit into the semester that they were planned to be taken.

After interviewing students to assess how they currently create their plan of study, I interviewed Michele Peck from the office of Academic Advising. Michele gave me insight into

how the University recommends that students create a plan of study. Her normal advising role is to help students who are undecided majors to determine which major to take. Not even the office of academic advising uses the current plan of study application when helping students. According to Michele, she and other advisors normally use paper and pencils to write out a student's plan of study, using information from major worksheets. She typically would write out several different major options for the student in order to compare them. She stressed the importance of students taking the time to take notes about their schedule: why are they taking this class? What requirement does it cover? She says that taking notes about a plan will keep a student from becoming confused about why they arranged their plan of study that way.

During the interviews I conducted, I asked what the students and staff would like to see out of a new plan of study application. They stated the ability to add minors, certificates, and other special programs to their plan of study would be important. The students interviewed also would like to see the ability to drag and drop courses from semester to semester. Michele Peck suggested a new system that could notify departments as to what students' plan of study had them taking for the next semester, allowing departments to gauge the demand for specific courses. Departments on campus could tailor what courses were offered each semester based on what students were planning on taking.

My interviews and analysis of the current system led me to draw some conclusions about the tasks that different types of users would be interested in completing using the new system. I created a high level task analysis that described the types of tasks that students and advisors would want out of the system. This document broadly described the different groups of users and how each group would view the system differently (See Appendix B: Plan of Study Application UI Requirement Analysis).

After analyzing the current system and interviewing the potential users to create a broad requirement analysis, I created a specification document to break down in detail the tasks users should be able to complete in the new application (See Appendix C: Plan of Study Application UI Specification). My specification document is listed as individual tasks a user should be able to complete and how the system should respond. The specification document gave a list of things that needed to be in my system and provided me with the information I needed to begin the wireframing process.

Wireframing & Design

Wireframing is the process of creating rapid layouts for the system and working through the structural design of the information architecture. My specification document gave the functionality that had to be included within my system, but before I could design how the system would accomplish those function I had to structure them into a design model that would fit the users' mental model. A mental model is how a user visualizes and conceptualizes a problem. Information architecture is how the information in a system is structured and organized when it is presented to the user. In order to match how I intended the user to conceptualize the application with how they would actually view it, I began by creating a good information architecture. Through the interviews I completed, I had a good knowledge of how a user typical completes the task of creating, modifying, etc their plan of study. To create good information architecture, I began by organizing the functionality in my specification document into categories based on the type of task that being accomplished. Those categories turned into menus, and groupings of menus whose functionality was related. My goal was to keep the user in mind when structuring my functionality. By continually asking myself where a user would look for a certain function, I attempted to bridge the gap between the design model and the user's mental model.

After organizing the functionality, I began to develop a wireframe layout using Adobe Illustrator. Illustrator's vector shape tools allowed me to quickly develop a layout without the software getting in the way. The program also allows for the wireframes to easily grow into more detailed layouts by adding colors, gradients, and stylistic elements. Color and other stylistic elements were purposely left out of the initial layouts so they would not distract from the information architecture and the organization layout of the interface. Keeping with just black and white layouts in the beginning keeps the focus on the usability and the functionality of the interface and not on the color palate or design style. Throughout the wireframing processes, styles and color were slowly added to enhance the richness of the interface as the structure and layout were solidified.

My first wireframes included a tabbed navigation that separated hypothetical plans from the active plan. This structure is similar to how the hypothetical plans and active plan are separated in the current system, but the current system forces the user to return to a home screen before selecting another plan to view (See figure 1.1). I implemented these tabs so that a user could easily switch back and forth between their hypothetical plans and their active plan.

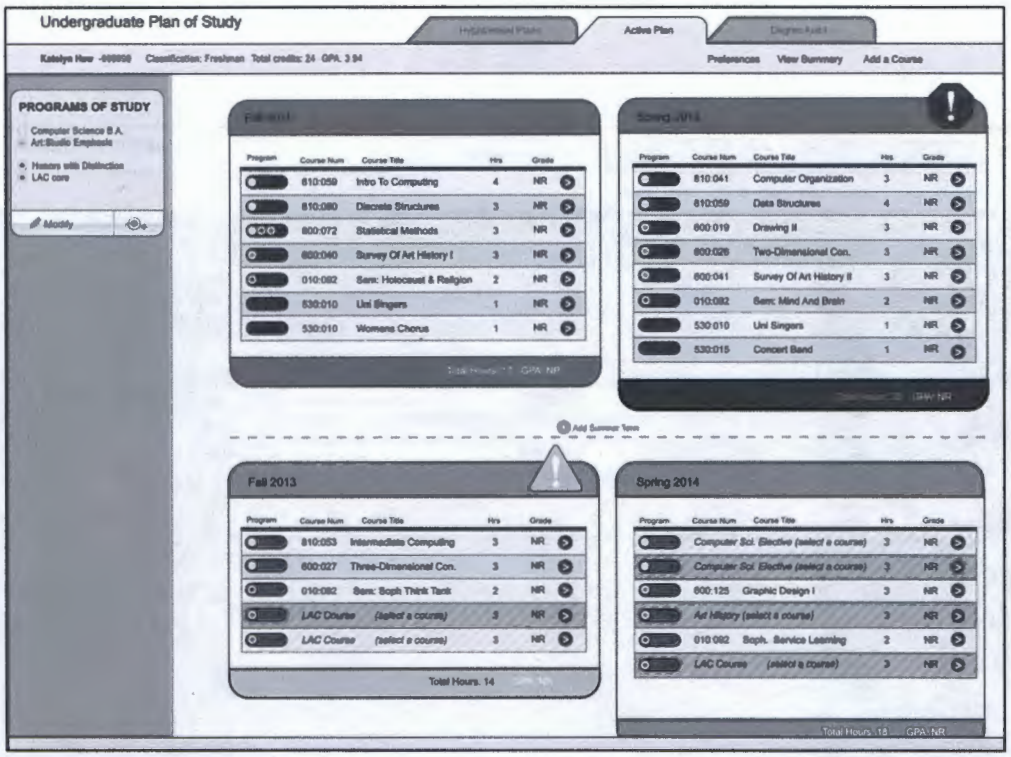


Figure 1.1. Viewing an active plan.

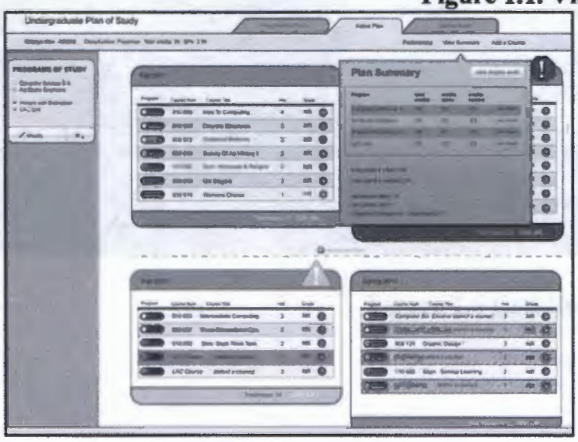


Figure 1.2. Viewing the summary of the active plan

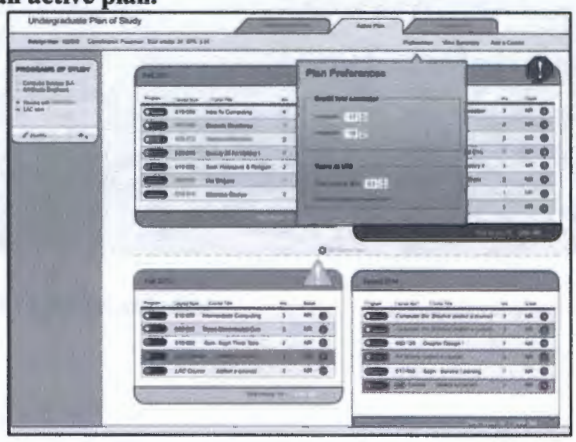


Figure 1.3. Editing the active plan's preferences

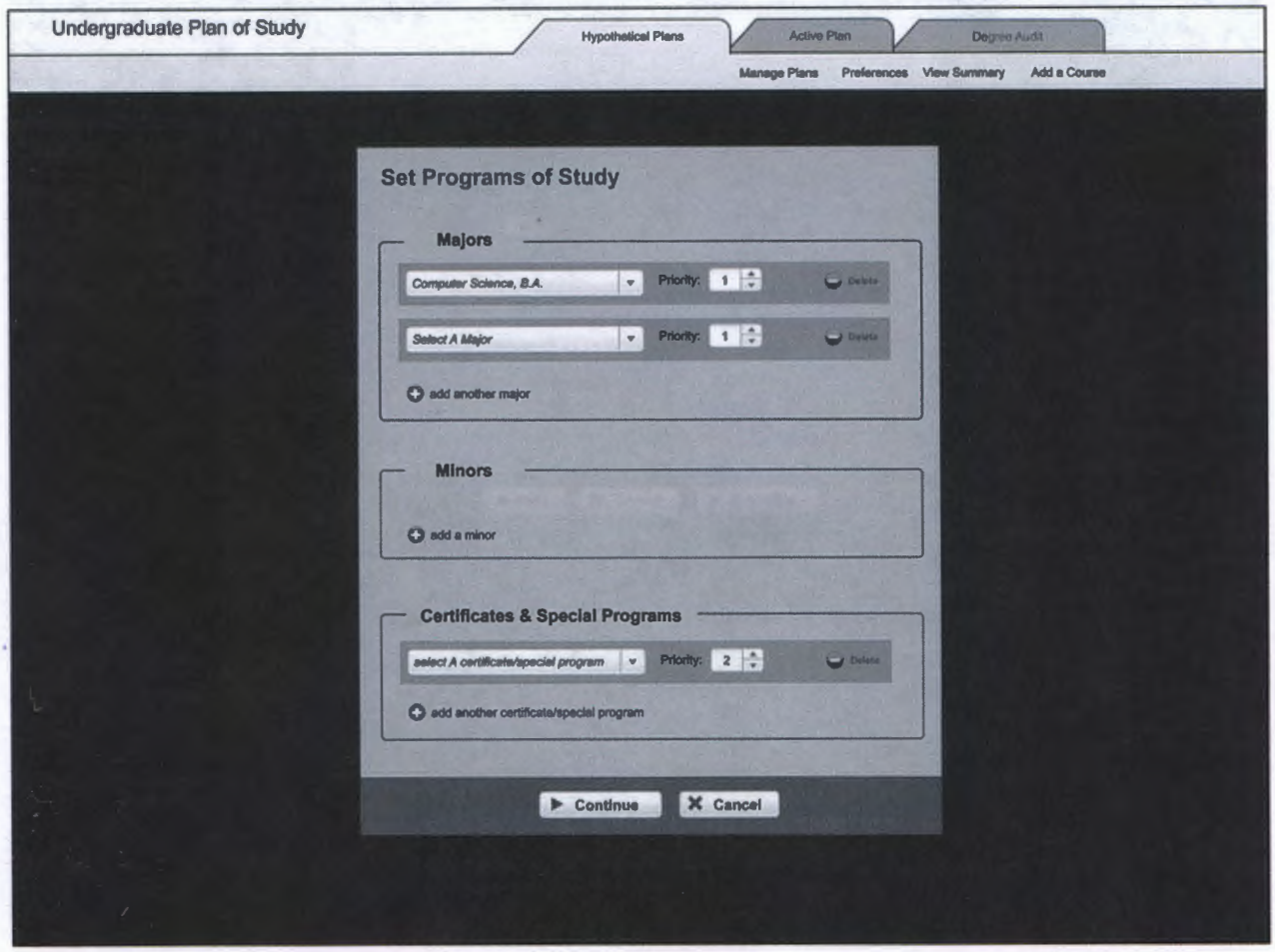


Figure 1.4 Creating a new hypothetical plan

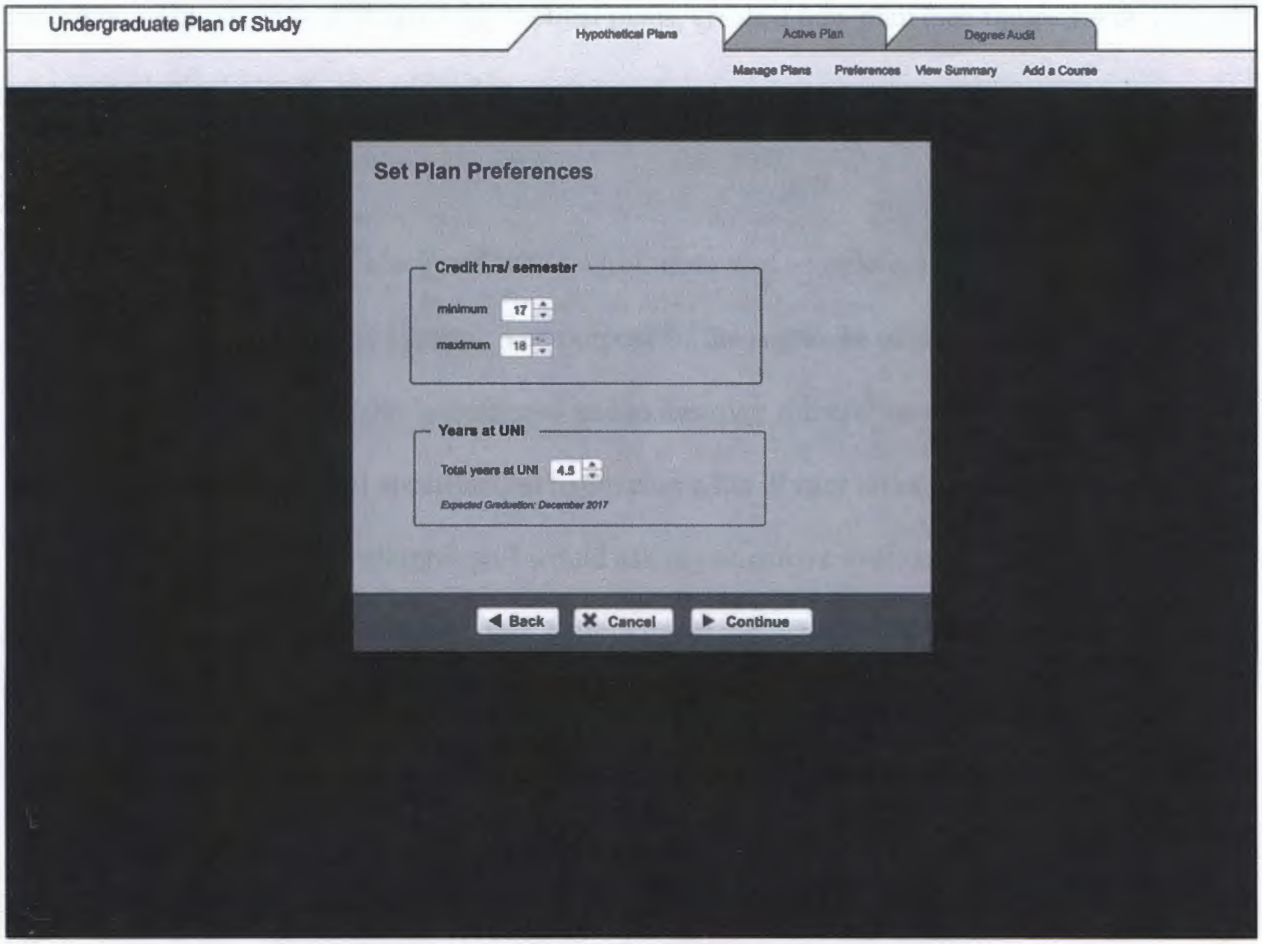


Figure 1.5. Creating a new hypothetical plan

These wireframes allowed for a user to modify their programs of study as well as their legend's color codes in a panel on the left sidebar. Functions that would not be used as often were placed at the top of the screen and included: "View Summary" (see figure 1.2), "Preferences" (see figure 1.3), and "Add a Course." Further analysis of this wireframe design would conclude that the "Add a Course" menu was out of place at the top of the screen, and it was moved in the next version. All of the other functionality for the system could be completed by clicking on objects within the schedule itself. The hypothetical plan tab contained almost an identical menu structure except it included a "Manage Plans" menu which allowed the user to

switch between viewing different hypothetical plans, create a new plan (see figure 1.4 & 1.5), or set a hypothetical plan to an active plan.

Cognitive Walkthroughs

After my first set of black and white wireframes was complete, I completed my own cognitive walkthrough of my system. The purpose of the *cognitive walkthrough* was to identify if I was missing anything in my wireframes and to discover if there were any major usability errors. I used my functional specification to develop a list of user tasks to use in the walkthrough. During the walkthrough I would ask my cognitive evaluator to explain how they would to attempt to accomplish each task by looking at the wireframes. This set of tasks became the basis for all of the other UI testing I completed on this interface, and they were all derived from the functional specification (See Appendix D: User Tasks for Cognitive Walkthrough). The walkthrough prompted me to change several items within the system (see figure 2.1). I discovered how the "Add a Course" functionality did not fit the rest of the menu items at the top of the screen. A user's initial reaction would be to look for that functionality on the side bar or in the plan itself. In this generation of wireframes, I chose to break the "Add a Course" function into three different types of course to add: "Program of Study," "University Elective," or "LAC Course" (see figure 2.2). This is how the current system allowed a user to add a course. With the basic layout and structure completed, I began to add stylistic elements and better define the look and feel of the system. Before doing cognitive walkthroughs with other evaluators, I fleshed out the interface and mocked up any screens I was missing (see figures 2.3-2.7).

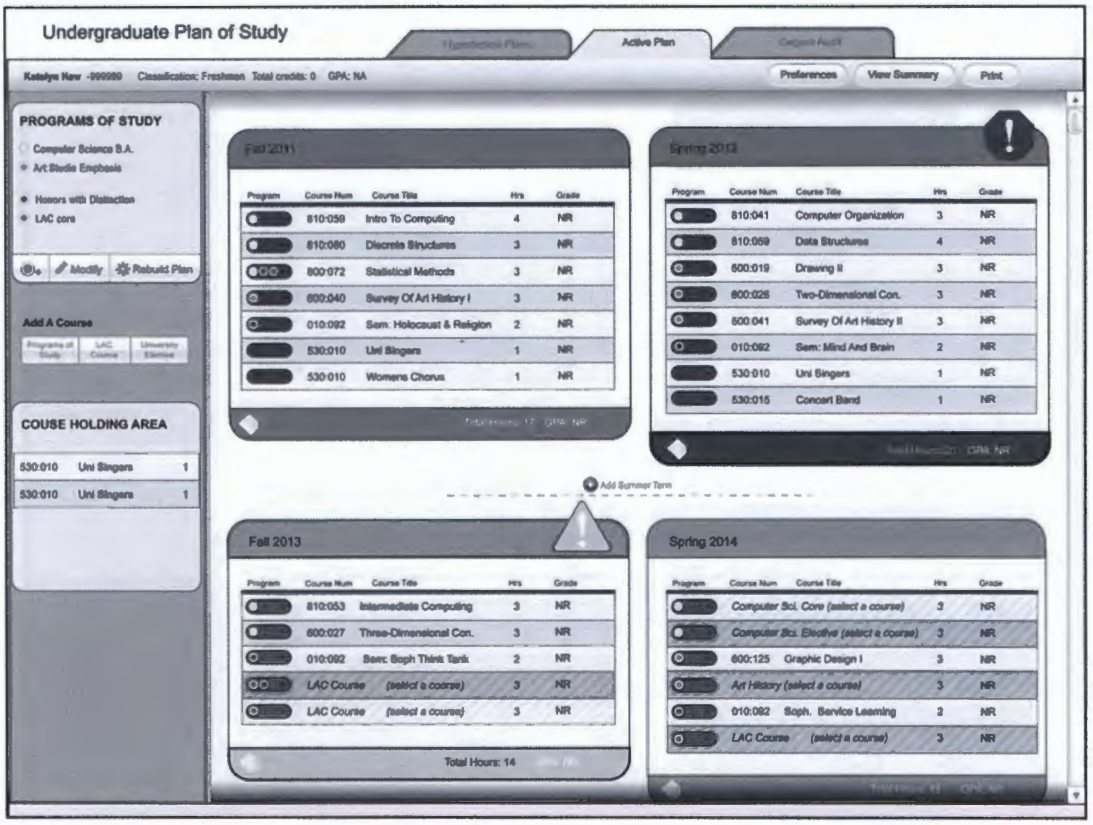


Figure 2.1. Viewing an active plan

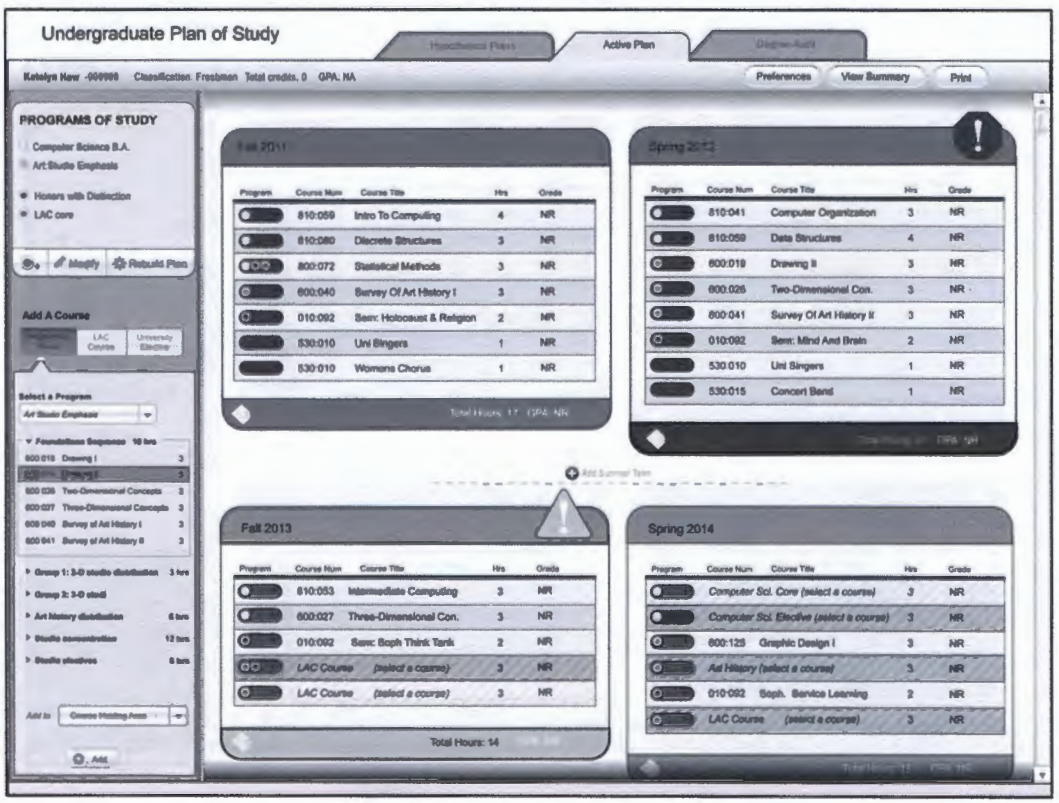


Figure 2.2. Add a Course from a program of study

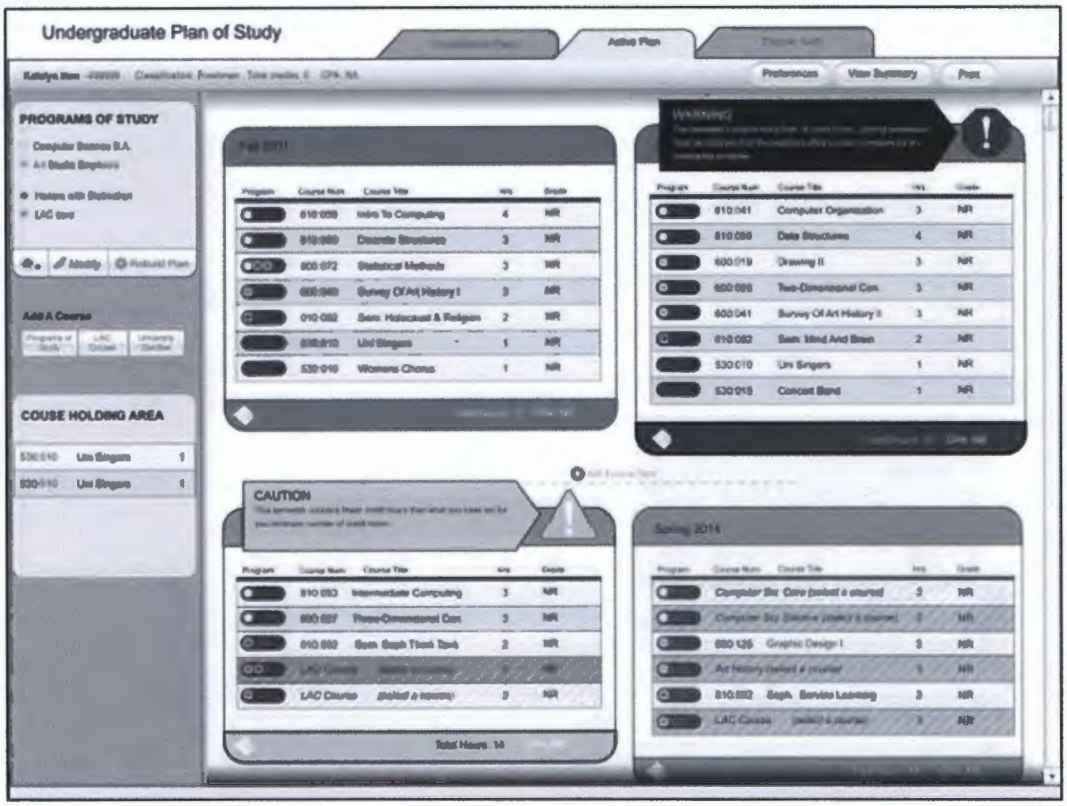


Figure 2.3. Viewing errors in the active plan

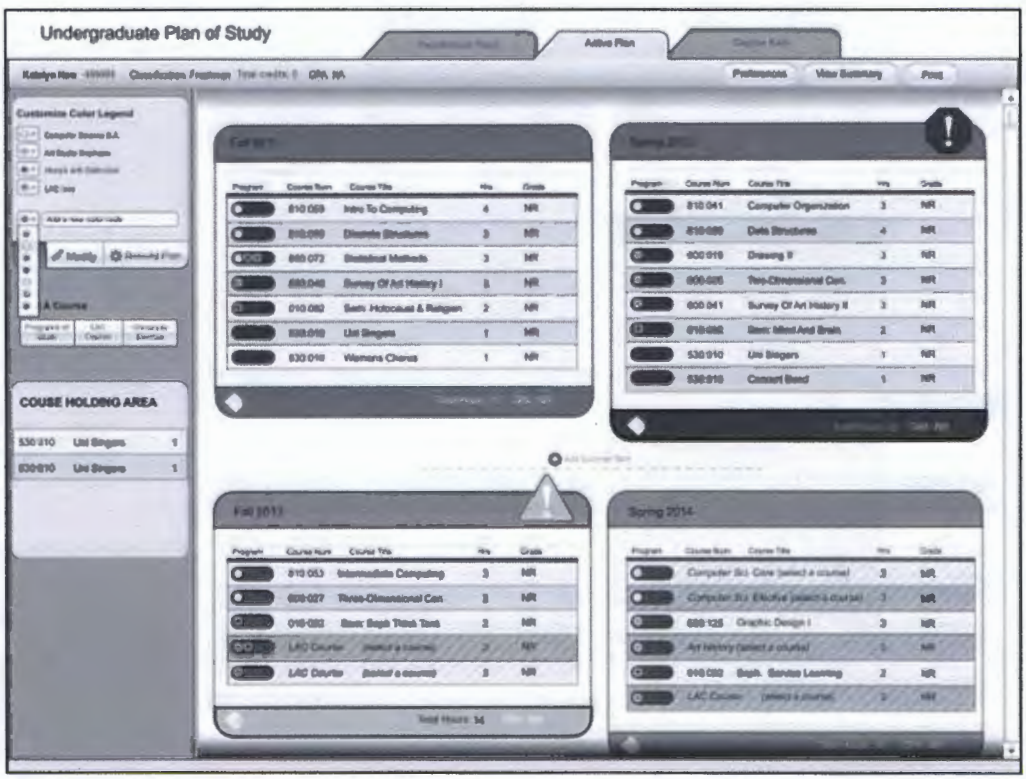


Figure 2.4. Editing the color legend

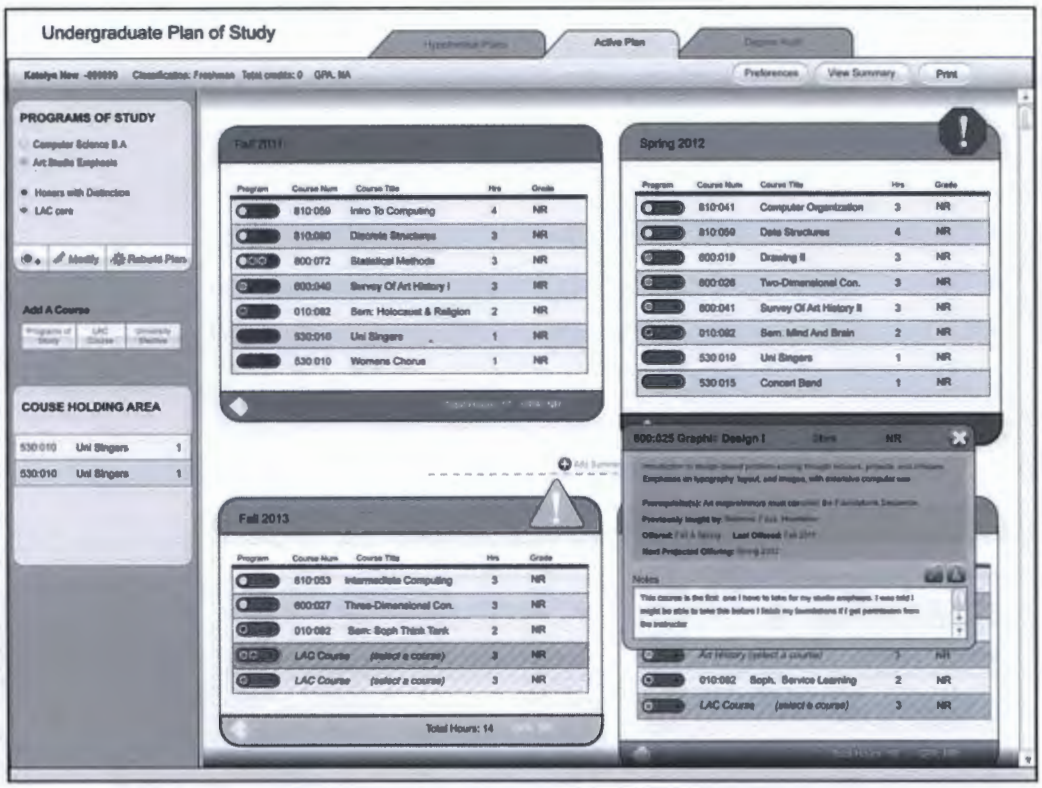


Figure 2.5. Viewing the details of a specific course

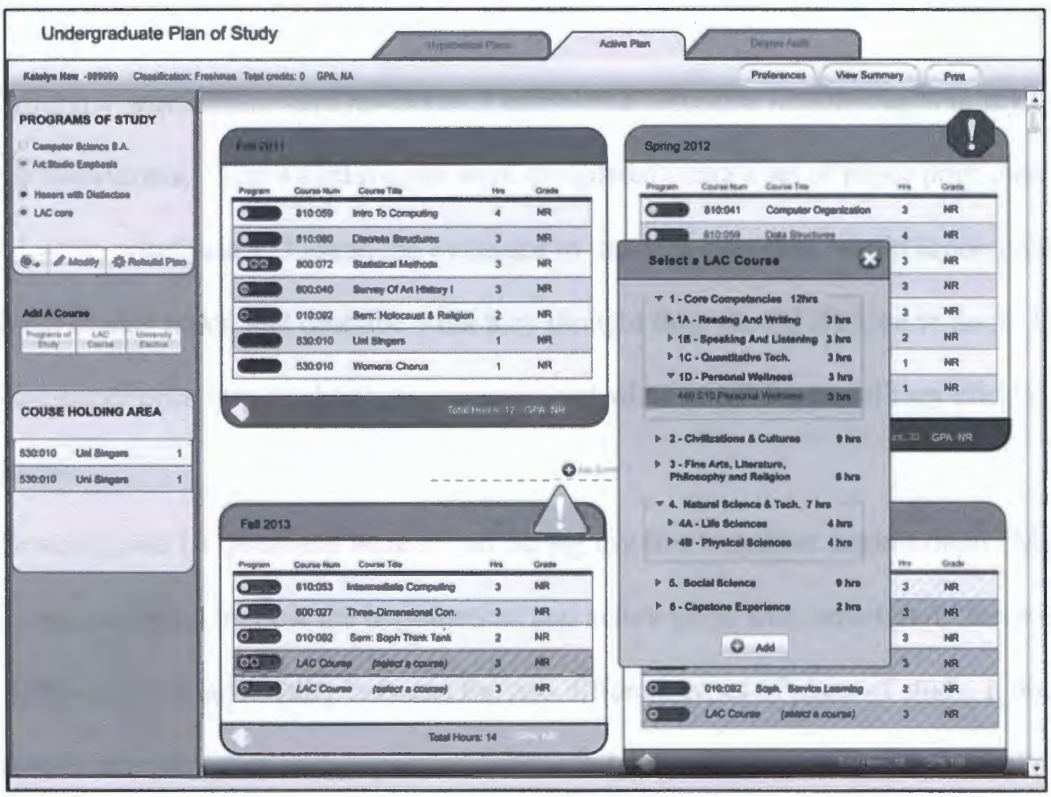


Figure 2.6. Selecting a specific LAC course to fill a requiremen

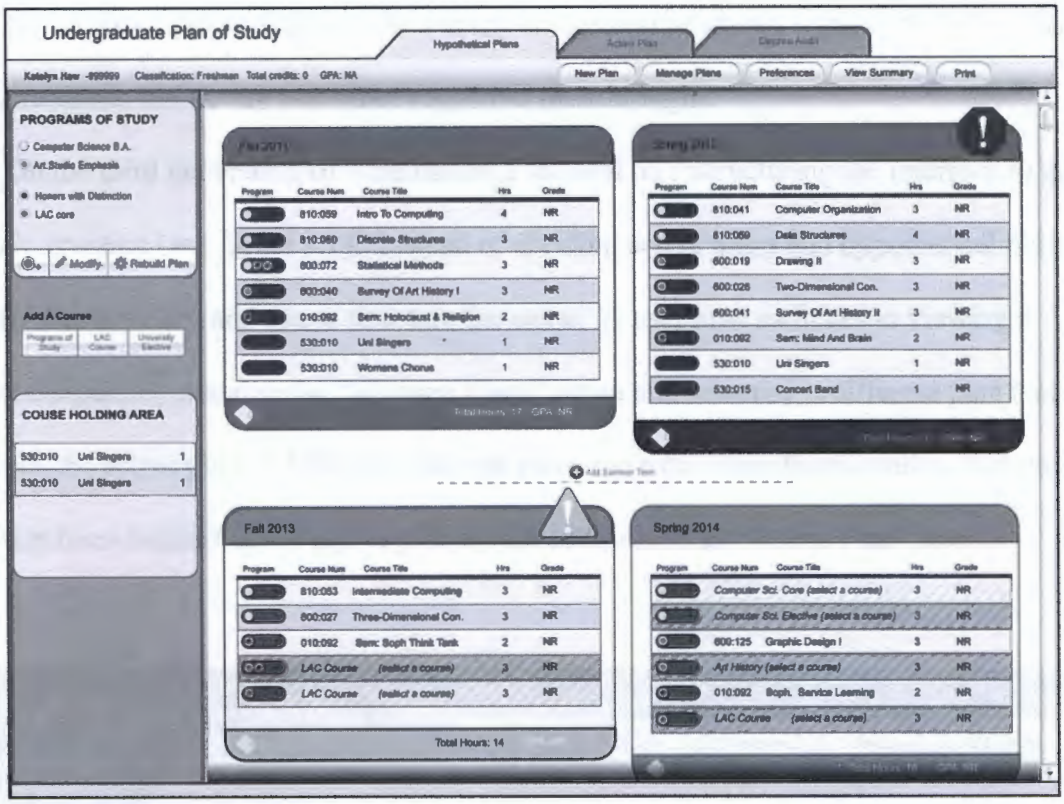


Figure 2.7. Viewing a hypothetical plan

After my wireframes were reworked, I completed *cognitive walkthroughs* of my system with three individuals. The walkthroughs were completed using a set of paper print outs of the black and white wireframes. I asked the evaluator to describe how they would accomplish a task from the list of user tasks, and describe what they thought they should click on to do it. I flipped through the paper print outs to show other screens and what would happen if they clicked on certain items.

Several good UI questions were raised during my first cognitive walkthrough. My evaluator was confused by how the hypothetical and active plans were structured. Because there was no difference in functionality between the two different types of plans of study, it did not make sense to him why they were separated functionally by a tab structure. Traditionally, a tabbed structure denotes a new section of functionality that is separate from the functionality of

the rest of the tabs. I reworked the flaws in my design and produced a third set of black and white wireframes before my two other cognitive walkthroughs.

On the third generation of wireframes, I focused on restructuring the interface to get rid of the tab structure (see figure 3.1). Instead of dividing active plans and hypothetical plans, I kept their functionality and menu structure the same. A user now switches to viewing a hypothetical plan by opening the "Manage Plans" menu and selecting a different plan (see figure 3.2). Both the active plan and the hypothetical plans have the same functionality; the active plan has merely been marked as the active plan by the addition of an "Active Plan" icon.

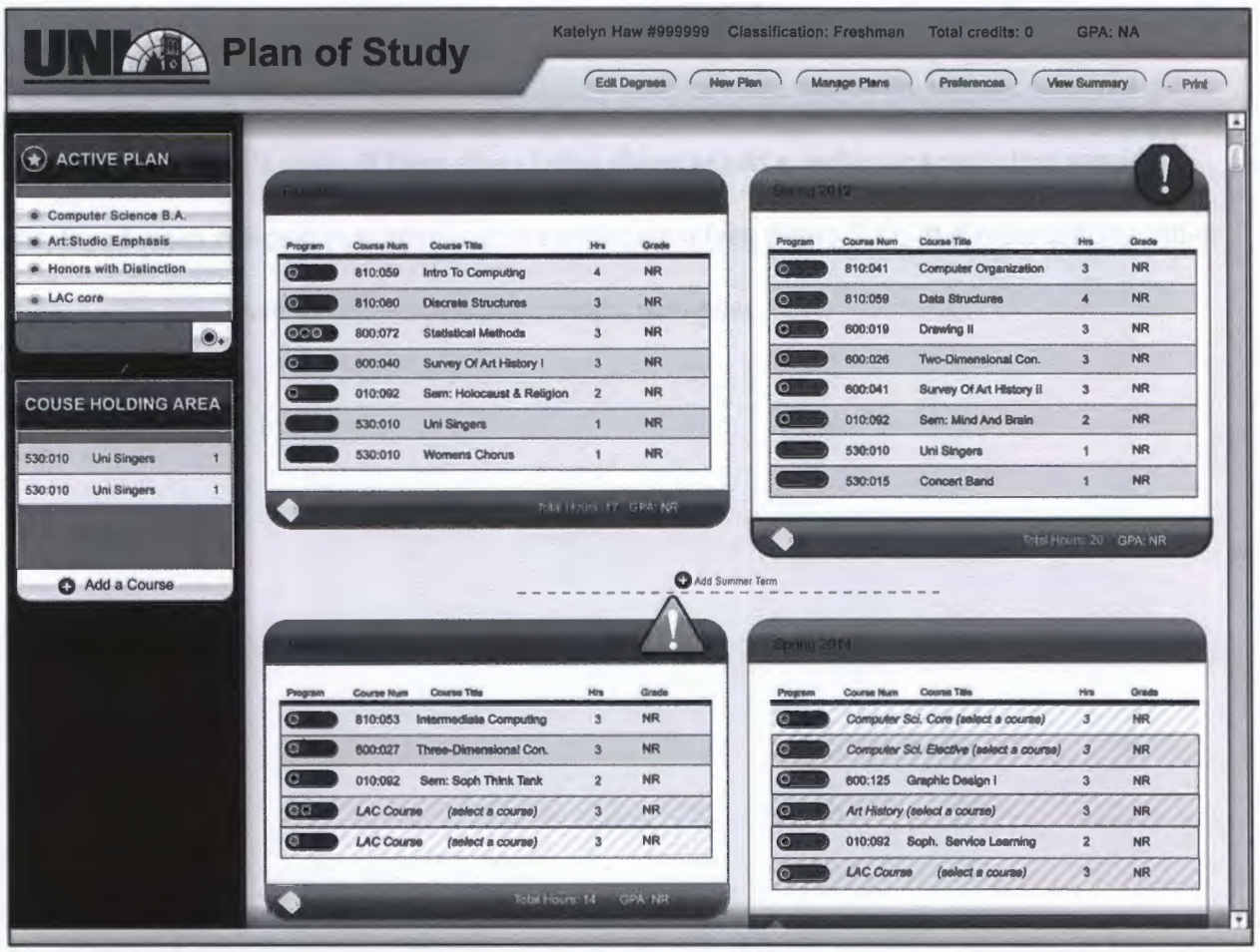


Figure 3.1. Viewing the active plan

The menu items at the top of the screen contain functionality that the user would not use often. I chose to move the functionality for editing the majors, minors, etc. to the top bar so that it could be close to the preferences, which also affects an entire plan. I chose to combine the functionality for adding a course into a single menu item, which is still located on the side bar, but now it is in connected to a course holding area (see figure 3.4). The course holding area allows the user to add a course that they are interested to the holding area before they add it to a specific spot in their plan.

During this generation, I also focused on increasing the contrast and developing a stronger hierarchy (see figure 3.5). The darker side bar encourages the user's eye to draw downward towards the plan of study itself where most of the commonly used functionality will be. The less used functionality is on the lightly colored menu bar at the top and does not get in the way of the user's view of their plan. I also chose to add a welcome screen that would be displayed when the user first arrives at the application (see figure 3.3). It directs users to either view a plan they have already created or create a new plan.

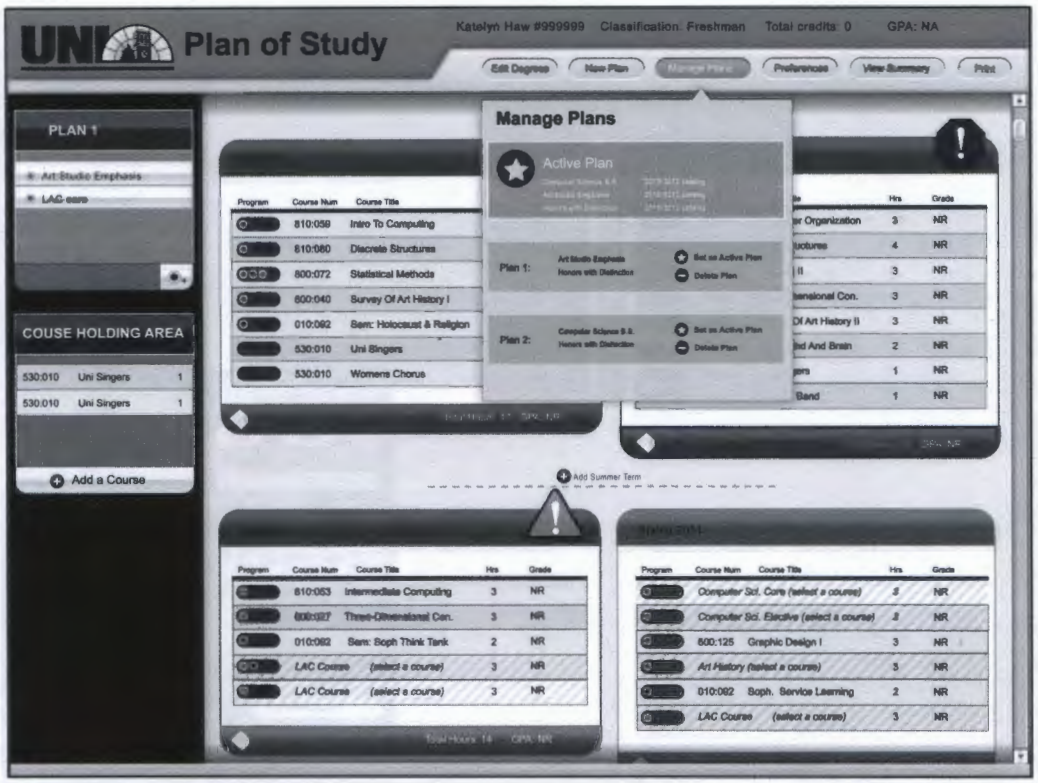


Figure 3.2. The manage plan menu allows you to switch between viewing plans as well as set a different plan as the active plan and delete a plan.

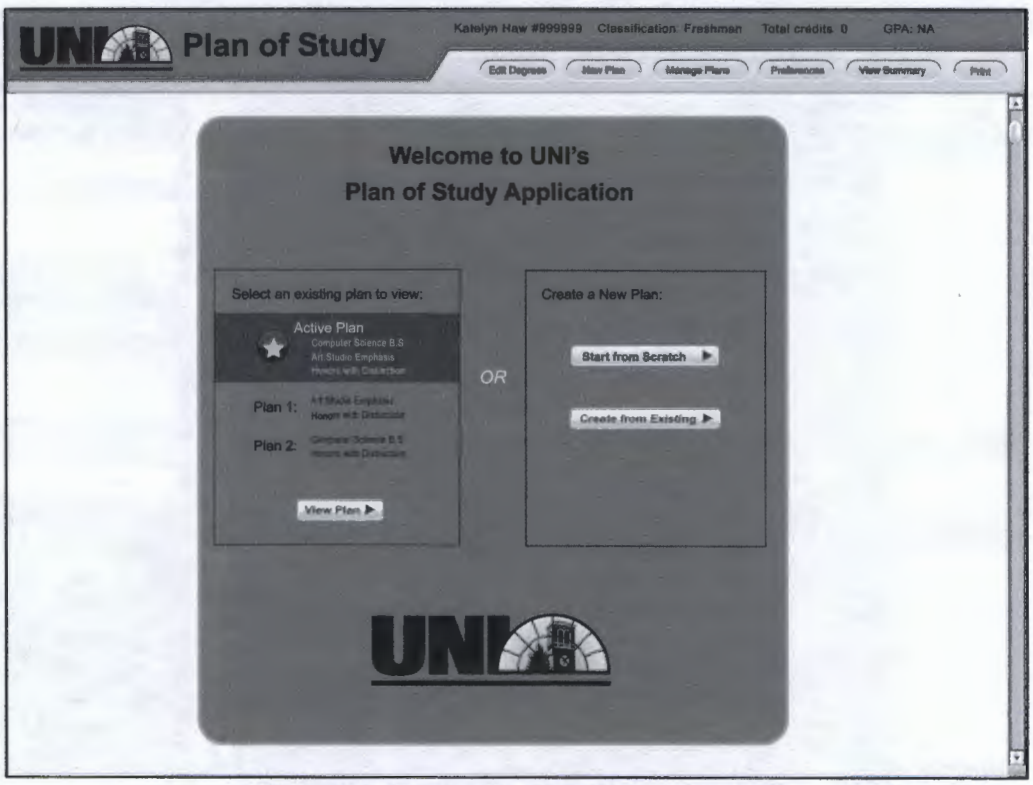


Figure 3.3. The Applications Welcome Screen

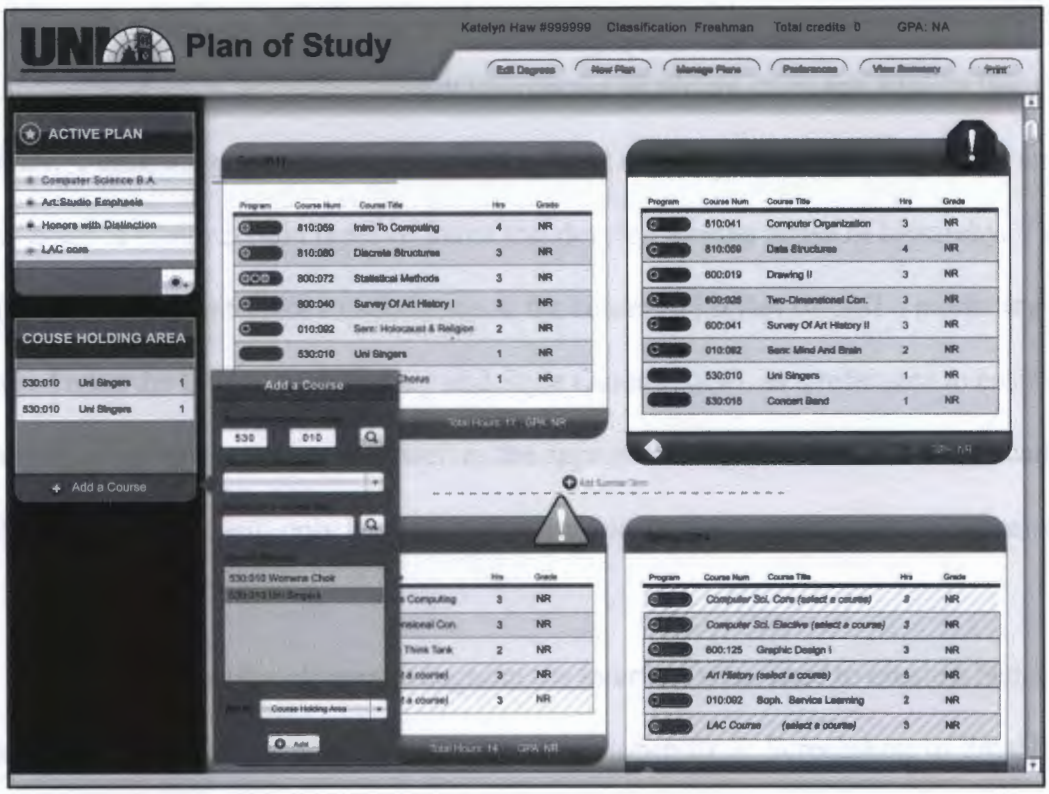


Figure 3.4. Adding a Course

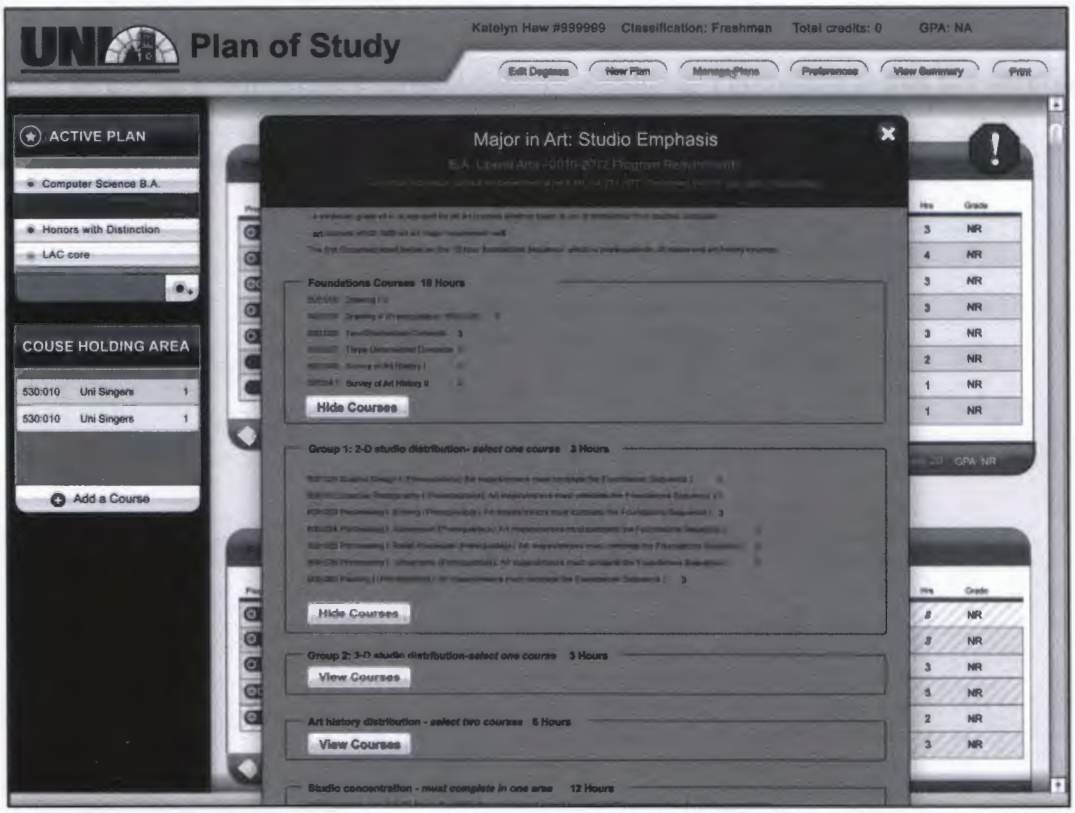
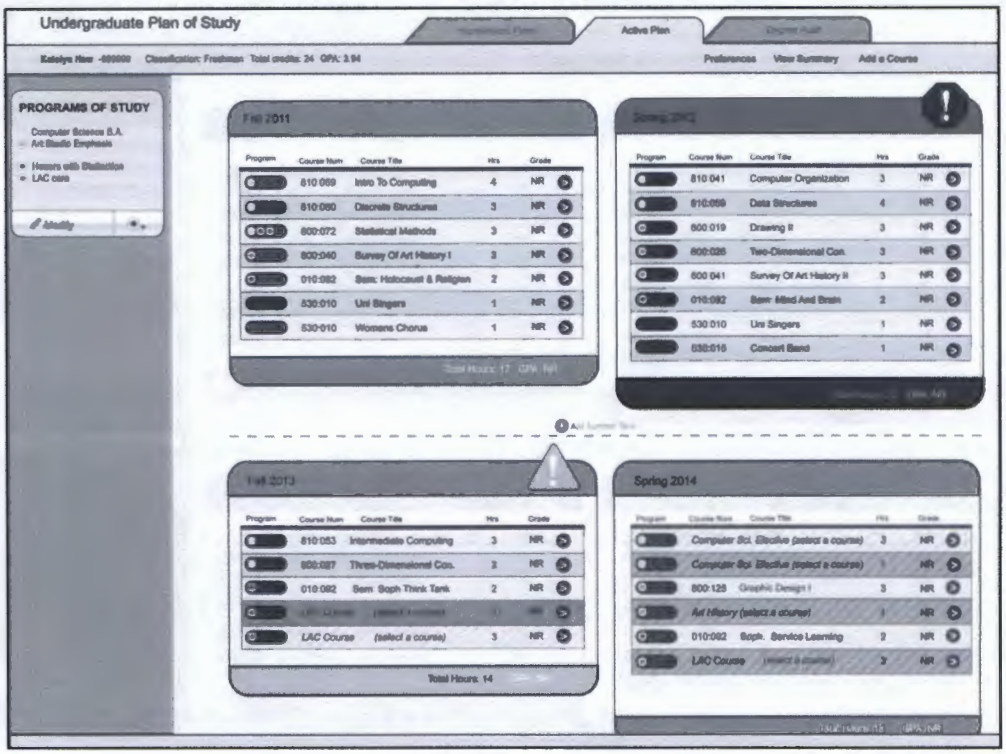


Figure 3.5. Viewing the requirements for a major

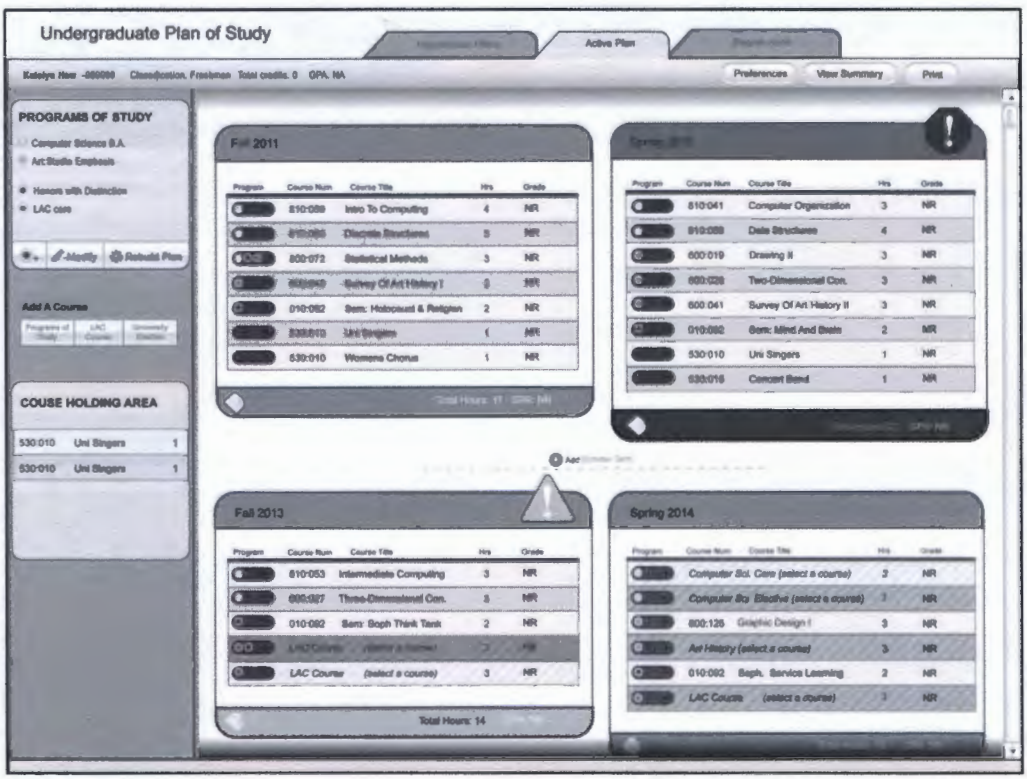
After my cognitive walkthroughs brought to light all of the major usability problems within my wireframes, it was time to begin the process of adding color and style to the wireframes. I used the greyscale wireframes that I had developed in Adobe Illustrator and began to add in color to group certain types of items and develop a better sense of hierarchy. I chose to keep my application's interface in adherence to the colors and styles of UNI's marketing and branding. I used the UNI Visual Identity and Style Guide (2010), as a reference to ensure I used the correct hues of purple and gold as well as the appropriate logo. Because my application is an internal application that will only be used by faculty and staff, I could use UNI's abbreviated logo.

The colors and styles chosen throughout the interface were used to create a better sense of order, hierarchy, and to suggest functionality (See Appendix E: Screenshots of Final Color Wireframes). By adding gradients, shadows, and other stylistic elements to buttons and other clickable or selectable items, the user is given a visual cue to its functionality. One of the advantages of creating wireframes using Adobe Illustrator or Photoshop is that the wireframes can be broken down into assets to be used in future prototypes.

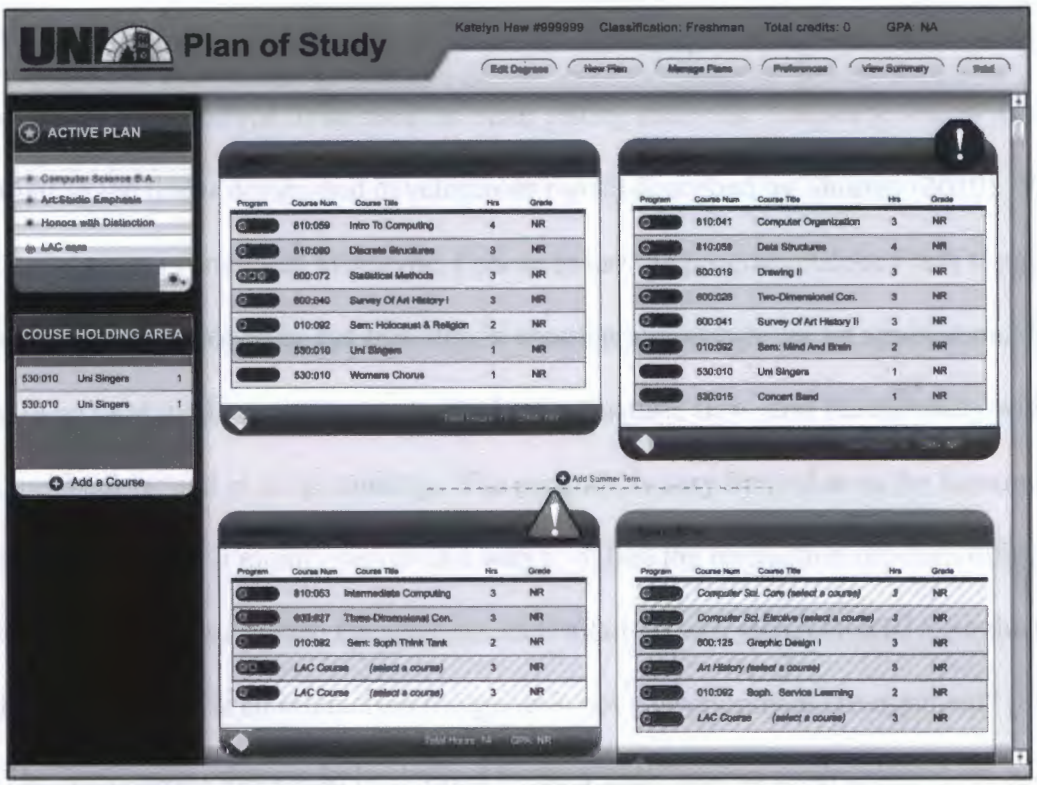
Wireframe Evolution



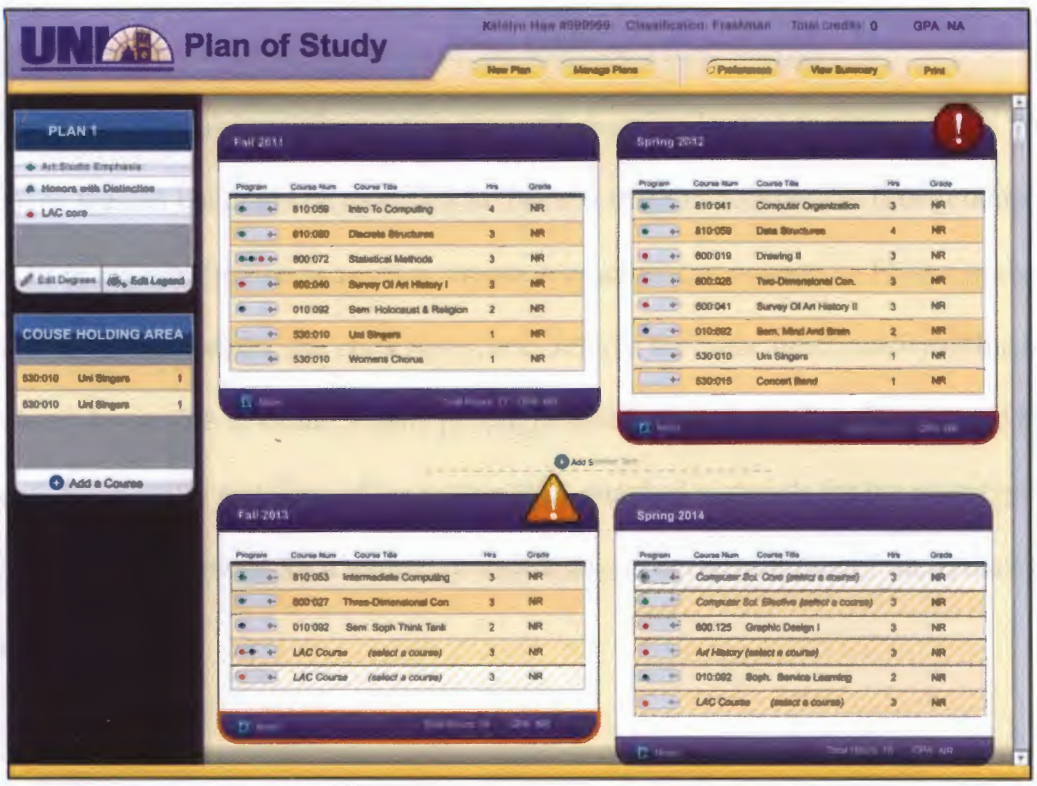
Generation 1: Initial black and white wireframes



Generation 2: Wireframes after my own cognitive walkthrough



Generation 3: Wireframes after my first cognitive walkthrough



Generation 4: Final Color wireframes

Prototype Production

Upon the completion of the full color version of wireframes, the system was ready to be implemented into a prototype to be used for other testing methods. I chose to follow a design flow similar to the linear design and development model described by Shorten (2010). For the production of a semi-functional prototype, I chose to use the program Adobe Flash Catalyst. Flash Catalyst is designed to be the first step in creating a prototype for an application. It is intended to be used by designers to create the basic structure, flow, and functionality without needing any background in programming. The program is very limited as to the functionality that can be created, and it mainly serves as a way to define the interaction between different elements of the interface. "Flash Catalyst has its limitations as a standalone RIA production tool, and quite rightly so—it is an interaction design tool, not a development environment" (Shorten, 2010). The interactions that could be defined by the Flash Catalyst program gave me enough interaction and functionality to do a heuristic evaluation on my prototype before too much time was spent on the specific implementation of the interface's functionality.

The benefit to this workflow is that the work done in Catalyst to create a basic prototype would not be wasted but could be modified in the next stage to become the final product. I used elements from my wireframes as assets to create interactive components in Flash Catalyst. I developed the rollover states, the click states, and defined the basic functionality of the buttons in the interface. I was able to create a basic prototype that a user could click through and discover the general functionally and flow of the system (See attached materials for the prototype). This level of prototyping allowed for a more thorough analysis of the usability of the system than the wireframes.

Heuristic Evaluations

The next step in the analysis of the user interface was a heuristic evaluation of the basic prototype. To prepare for the heuristic evaluations of my basic prototype, I modified an existing heuristic checklist that was based of Jakob Nielsen's 10 heuristics for usability (Nielsen, 1994). Because this heuristic checklist was meant for software that was complete and fully-functional, I was able to eliminate several items because they were not applicable to the current prototype. I created a document which included the heuristic checklist, basic instructions on heuristic evaluations, and a description on how to access the online demo. Because most heuristic evaluations suggest using 3-5 evaluators with background knowledge in usability and interface design (Nielsen & Molich 1990), I chose 3 evaluators with moderate to advanced usability expertise.

My evaluators consisted of two professors who have either previously taught a user interface design course or have done research in the field of usability and interface design. My third evaluator was a graphic designer who has recently been actively involved in several user interface design projects. I found benefit in having evaluators with experience in the development as well as design because it provided a breadth of comments that encompassed both the aesthetic and usability of the interface. The professors' backgrounds in development led them to focus on the functionality, terminology, and architecture of the system. The graphic designer was able to provide important feedback about the aesthetic appeal, the layout, and the page hierarchy of the system.

Several important usability concerns were discovered upon reviewing the evaluators' ratings and comments. Two out of the three evaluators were concerned with the use of the term "degrees" to encompass all areas of study including majors, minors, and certificates. In early versions of the interface, I had referred to these items as "programs," but a suggestion given

during an early cognitive walkthrough prompted it to be changed to "degrees." Further analysis of problems like this where mixed feedback is received can take many forms. Polling a sample user group would most likely provide a more clear understanding of what a user would find most familiar and understandable.

One evaluator was confused by the combination of the legend with the listing of degrees being achieved. He said it "is confusing" and the interface "should separate degree/major stuff & legend stuff." This was feedback that had not been received previously. Although the previous cognitive walkthrough evaluators easily understood the combination of the legend with the listing of majors, the new feedback brought up a previously unexplored situation. Because the system allows users to add their own color code to the legend to denote something the users chooses, how are those color codes visually separated from color codes that specifically denote degrees being achieved?

Although the system being evaluated was a partially functioning prototype, the heuristic evaluations brought up new usability questions about the interface. Heuristic evaluations are highly critical in nature, because the checklist encourages the evaluator to review every aspect of the interface in multiple ways. The feedback from this evaluation highlighted areas in need of further usability scrutiny. The feedback from the heuristic evaluations led to another iteration of revisions in the wireframes and basic prototype.

User Testing

The scope of my thesis project was not able to include the final implementation of the system or end user testing due to time constraints. Once the system was fully functional, the next step in the design process would be a set of user test on a final version of the interface. Final users tests would ensure that the user can easily learn the software and can fully complete all of the tasks in the system. Before user testing could begin, the user's tasks that have been used in

previous evaluations would be broken down into specific steps the user would take to complete the task, including any input data they may require. The system would be tested with a set of users from the group of people who would be potential end users of the application. The users would begin their testing session with a short interview with the test administrator so basic background information could be collected. The user would then try and complete the tasks given to them by the administrator on their own. The administrator would not interfere or direct the user unless they were extremely confused or ended up too far off track from the task at hand. A video tape of each user test would help in the later evaluation of the interface to see exactly where the user struggled to accomplish a task. At the end of the user test, the user would complete a short questionnaire with questions about their experience. These questions would help determine the success of the interface. After any further corrections or modifications to the interface and backend, this application would be ready for final error testing and deployment.

Limitations

The primary limitations of my thesis project were the lack of time and resources because I was attempting to accomplish something that is normally done by a team of people working full time for 6 months to 1 year. Because the goal of this thesis was to explore all stages of the design process, not all stages were executed as thoroughly as would be expected with a full team of designers and developers building a full application.

Analysis & Specification Limitations

I limited myself in the amount of problem analysis I completed because of the time constraints and the small scale of the application I was trying to complete. Due to time constraints I was not able to interview a representative sample of students to fully gauge all of the requirements students may have for the system. A more in depth research and analysis

would have included five to ten student interviews instead of two. In the design cycle of a larger system, I would recommend more analysis be done on the types of users and the different methods that could be used to compile an optimal schedule. These different methods of compiling plan of studies could be used to create different mental models. These mental models could then be compared to determine how the application could best be structured to match how most users view and understand the problem. User stories could be created to describe fictional users, their circumstances, and they desire for the system. One of my largest recommendations for early requirements documentation for any larger system would be the creation of detailed use cases. These use cases would fully describe, click-by-click, how a user would complete a task and how the system would prevent errors and handle errors. Since the field of User Interface Design is newly developing, I feel that a thorough investigation of the how the users currently complete the tasks the new system will assist them with will greatly enhance the usability of the final system. In particularly complex domains, this analysis can mean the difference between a system's success and failure.

Cognitive Walkthroughs Limitations

After completing three cognitive walkthroughs with the black and white wireframes, I found that it was often difficult for users to interpret the meaning of items on the wireframes. It was difficult to flip through paper print outs of theaw wireframes as the evaluator chose something they would click on to determine an action. This created an awkward flow to the walkthrough that was disconcerting to the evaluators. For future cognitive walkthroughs use of a .pdf document would be a better choice than of paper printouts. Doing the cognitive walkthroughs on a screen would help the user develop context for the interface as well as make it easier for the designer to jump from one page in a .pdf document to another.

This design process suffered from a lack of a small group critique of the wireframes at an early stage. For larger projects with a series of designers working on the same system, I would recommend a small group cognitive walkthrough as the first walkthrough to take place. This would allow the designers to discuss and debate the best information architecture and mental models for the system. By following a group walkthrough of the system with individual cognitive walkthroughs done by evaluators not in the previous group, both large structure-orientated and small detail-orientated problems would be brought to the designer's attention at an early stage.

Prototype Production Limitations

The prototype production stage was primarily limited by the current technology and software available. Since the field of user interface is moving more and more towards an interdisciplinary process where designers and developers are working more closely and concurrently in order to produce the most usable product in the shortest amount of time, the interface designing tools being developed are also attempting to bridge the gap between designers and developers. Software that is meant to give designers with no background knowledge in computer programming the ability to produce prototypes are increasing, however; there are many drawbacks to the rapid prototyping software available today.

For this project, I chose to use Adobe Flash Catalyst to design the systems' interactions and the basic prototype. I chose this software because of its integration with the other Adobe products and the ease of creating basic interactions. It could easily import the wireframes from Adobe Illustrator and would create a prototype that would not go to waste but be used in Adobe Flash Builder to create the final interface. In theory, the Adobe design flow from Illustrator to Flash Catalyst to Flash Builder is an ideal one which makes close ties between the designers and

the developers. However, I discovered many flaws to the current system which became major limitations to the development of my prototype and the final interface.

Adobe Flash Catalyst is relatively new program to the Adobe Suite making the program less refined than Adobe's other products. Catalyst uses an object oriented approach to building an interface by allowing the designer or developer to create reusable components. These components can be customized from Catalysts toolbox or can be created using a custom component. The tool kit within Catalyst is extremely limited, offering ways to create text inputs, buttons, scrolling components, and data lists but not dropdown menus, accordion menus, or other components that are standard to other web toolkits. The lack of these basic components requires the designer to either leave those elements as static images within the basic prototype or attempt to create a custom component that could replicate some of the functionality. Through the prototyping process, the development was significantly slowed down by the lack of certain components in the Catalyst toolbox or the absence of properties that my interface required be changed. Other than Catalyst's data list component, Catalyst does not offer allow any variables to be added to the interface. This limits a designer or developers ability to develop the interactions of the interface in Catalyst. It also limits the developer's ability to reuse certain components. I also found Catalyst to be an unstable development platform and was plagued by repeated crashes and file corruption. Despite having gone through several beta and final versions, Flash Catalyst still suffers from large bug problems and performance issues. This made my interface system too taxing for the software to handle.

In addition, the transition between Flash Catalyst and Flash Builder was not as smooth as the Adobe's workflow suggest. The catalyst program produced code that was overly complex and not neatly coded. Although it produced working code, it would not be code that a programmer would appreciate working with. A developer is most likely to completely rework

the code produced by Catalyst so that it was easier to read and work with. In the end, the time that was saved by the production of a prototype that could be turned into the final application would be negated by the time it took to rework the Catalyst demo into workable code.

Instead of rapidly producing a prototype that was functional enough to complete Heuristic Evaluations, I invested large amounts of time into producing a prototype that was full of hacks and bugs. For smaller systems with less complicated interactions, Adobe Flash Catalyst is still a good option for rapid prototyping for initial testing, however; the early version of this program lack vital components and the performance to handle larger, more complex interfaces. If future development of the Catalyst software improves the toolset and performance it would allow the Adobe Suite to offer a workflow from design to development that would increase usability of the final software and speed up the production process by integrating the roles of the designer and developer.

Heuristic Evaluations Limitations

Due to the lack of available UI experts, I chose to have three reviewers perform a heuristic evaluation of my interface instead of five which is the recommended limit. I believe adding two more reviewers would have helped to clarify areas on the evaluations where three evaluators disagreed or had conflicting answers. It is also recommended that the evaluators get together to discuss and compare their individual answers in a small group. Since my evaluators were volunteers, I could not require them to get together and spend the additional time to draw a group conclusion about their evaluations. In a future study it would be beneficial to have evaluators discuss their answer in order to draw a conclusion about the areas of the interface that require the most amount of work.

In addition, the prototype used for heuristic evaluations was not a complete functioning prototype. The evaluators were not able to fully evaluate every item in the heuristic checklist

because of they were evaluating a prototype. It would be beneficial for a system to receive another set of heuristic evaluations after the final functioning application has been produced. This would allow the evaluators to more thoroughly explore the user's interactions with the system. This additional set of evaluations should take place enough time before the final user tests so that the developers would have enough time to make corrections before the user tests. Despite the limitations created by going through this process as an individual completing the research, design, and evaluations in a single semester, this thesis remains a success exploration into the user interface design process.

Summary

I chose to do a study in the user interface design processes as a final culmination of my double majors. User interface is a unique crossover between design and development. The field of user interface design is newly developing and created a unique challenge and room for discovery of a better usability design process.

Expanded Discussion of Significance

The technology one uses in his or her daily life is rapidly expanding. Common tasks like banking, shopping, or doing taxes, that used to be done through paper-pencil methods are now being completed on the web, a mobile device, or through desktop software. As individuals are required to use technology for more and more tasks, it becomes important that users of all genders, socioeconomic backgrounds, cultural backgrounds, and technological backgrounds be able to use the same software and technology with as few errors as possible.

In the world of fast-paced software development, usability is often overlooked. Although the usability and user's experience of a piece of software can determine its success in the market place, often times companies do not devote enough time or money is on the development of the

user interface. The aesthetics and layout of a web application play a large role in whether a user chooses to use the application or if they will move on. According to Chen (2009), users make judgments about websites in the first 50 seconds of their entry to the site. The more aesthetically pleasing a web application is the more it will draw the attention of users, and the more successful it will be. However, usability is more than just beautiful aesthetics, having correct functionality, ease-of-use, lack of errors, and system feedback are all essential to creating a usable system. In fact, according to Chen (2009), research has shown that aesthetics help create an emotional connection to the interface but usability was found to be more important that aesthetics in creating a positive experience for users. This illustrates the need for both designers and developers to put the user experience at the forefront during the design process.

Description of Final Work

The result of my honors thesis is a fully developed interface that has been thoroughly tested for usability with a complete set of usability tests. During the design process I have created a requirements analysis, a functional specification, a series of wireframes, and a flash catalyst prototype. My interface underwent cognitive walkthroughs, heuristic evaluations, and user testing to insure its usability. The interface, prototype, and specification documents describe the look, interactions, and functionality of the web application. These deliverables are ready to be turned over to developers so that they can be implemented in the final web application.

The resulting interface is a well-designed improvement upon the existing system. This interface and accompanying specification documentation describe a system that would better fit the needs of students at the University of Northern Iowa who are struggling to create a plan of study. It would provide a more intuitive structure and layout than the current system. It would handle and prevent the user from creating errors in their schedule such as taking too many credits

in a current system. Most importantly it would provide the flexibility and customizability for students that the current system lacks entirely.

No user interface can guarantee that it is 100% free of usability errors. Although my current interface has been scrutinized and revised throughout several evaluations and tests, further usability problems could arise during the final implementation of this interface. Since usability is both the layout and structure of an interface and its functionality, more usability tests on the final application could produce errors in the functionality or how the interface and functionality interact.

Reflections on Experience

I have developed a better understanding and awareness of the difficulties of user interface design. My biggest concern for larger projects is the difficulty of communicating important aspects about the interface amongst all members of the design and the development teams. As an individual who completed the research and analysis, wireframes, and interaction prototype, I had a full understanding of the goals, scope, and style of the interface. In a larger project, the designers would have to be able to fully communicate their ideas and goals for the look and feel of the interface to the developers who would create it. In return, the developers would have to fully communicate the technical limitations of the system so the designers could create an interface which fits within the system performance abilities. Traditional software development relies upon specification documents to relay important information throughout the development team. The terminology in traditional documentation may not be friendly or may contain terminology specific to programmers and developers that may not be understood by a design team. I believe the TCUID design process can help designers and developers bridge the communication gap by creating specification documentation that focuses on the user and the tasks which should be able to be accomplished with the completed system. By utilizing the user

as a common vocabulary, the designers and developers can better communicate the goals of the system and its interface.

With the advancements in interaction technology like touch screen and motion detection technology the designer's job has become more than just designing the look of an interface. Designers now have the ability to design the motion and interactions that an interface has when responding to a user's action. Software such as Adobe Creative Suite's Flash Catalyst allows designers to design the interactions of an interface. Flash Catalyst and Flash Builder allow designers with little or no programming background build basic systems for prototypes or as final software. They provides a layer of abstraction from the code and contains a toolbox with components that will speed up the interface's production time. I believe software like this Adobe pairing is the future of interface and interaction design as the roles of designer and developer are merged.

Throughout the design and development process I have gained valuable knowledge and firsthand experience in user interface. Each step in the design process offers a different set of challenges that can only be discovered through a thorough exploration. The experience of having a part in every step in the software design process is not one that can be had during normal University coursework. Even when the software design process is taught, the process is rarely focused on usability and the experience of the end user. The interdisciplinary nature of interface design also makes it a difficult subject to thoroughly teach within a single course. Although the Computer Science Department at UNI offers a User Interface Design course, it is a course aimed solely at interface design from a development standpoint. My thesis work allowed me to expand upon that coursework and aim my explorations at the objective of a graphic designer and a developer. I was able explore deeper and more expansively than the limitations of a University course would have allowed.

Conclusion

Usability is not only difficult to define but the development of a usable interface requires skills from widely different disciplines that each have a different view of the design process. Designers often look at the design process as a fluid, creative process that cannot be constrained in a strict timeline. Programmers and developers view the design process as a specific set of sequential steps to develop software from the high level requirements and architecture to the low level implementation details. Interface design requires aspects of both the creative and development process yet the integration of the two and the communication between designers and developers can often be difficult. The TCUID process bridges some of the communication gaps between designers and developers by focusing the design documentation and process on the needs and tasks of the user. By describing everything in a design process in relationship to the end user, designers and developers can find a common language and goal from which to develop. New software such as the Adobe Creative Suite, offers an opportunity for designers to step outside traditional roles and become interaction developers without needing a background in programming.

As the field of user interface design continues to evolve and develop as more and more tasks become computerized, the roles of the designers and developers will continue to merge. Bridging the gap between designers and developers creates better opportunity for the advancements in user interface design. The look of an interface will better match the interactions and the functionality the back end software provides. This will open up a new era of interaction technology as the creative minds and the technical minds converge to create new products that will change how technology interacts with the world.

References

Bowen, J. & Reeves, S. (2009). Refinement for user interface designs. *Formal Aspects of Computing*, *21*(6), 589-612.

Buxton, B. (2007). *Sketching user experiences: Getting the design right and the right design.* San Francisco, CA: Elsevier.

Chen, J. (2009). The impact of aesthetics on attitudes towards websites. *Usability.gov*, Retrieved from http://usability.gov/articles/062009news.html

Eaton, E. (2003). *Designing web site interface elements: For graphic designers.* Gloucester, MA: Rockport Publishers.

Garrett, J.J. (2003). *The elements of user experience: User-centered design for the web.* New York, NY: American Institute of Graphic Arts.

Hussain, T., (2005). *Projections of federal tax return filings: calendar years 2005-2012* Retrieved from http://www.irs.gov/pub/irs-soi/12proj.pdf

Lewis, C. & Rieman, J. (1994). *Task-centered user interface design: A practical introduction.* Retrieved from http://hcibib.org/tcuid/

McCracken, D.D. & Wolfe, R.J. (2004). *User-centered website development: a human-computer interaction approach.* Upper Saddle River, NJ: Pearson Prentice Hall.

Molich, R., & Nielsen, J. (1990). Improving a human-computer dialogue. *Communications of the ACM*, *33*(3), 338-348. doi: 10.1145/77481.77486

Nielsen, J. (1994.). Ten usability heursitics. *useit.com*, Retrieved from http://www.useit.com/papers/heuristic/heuristic_list.html

Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. *Proceedings of the Conference on human factors in computing systems proceedings of the sigchi conference*

on human factors in computing systems: empowering people (pp. 249-256). New York: ACM http://www.acm.org. doi: 10.1145/97243.97281

Norman, D. (1988). *The design of everyday things*. New York, NY: Basic Book.

Program of Study Coordinator (2008). *Undergraduate Plan of Study.* Retrieved from University of Northern Iowa website: http://www.uni.edu/pos/

Roberts, D. (2010). Usability and mobile devices.*Usability.gov*, Retrieved from http://usability.gov/articles/newsletter/pubs/052010news.html

Shorten, A. (2010, April 30). Exploring flash catalyst and flash builder workflows. *Flash Catalyst Development Center*, Retrieved from http://www.adobe.com/devnet/flashcatalyst/articles/flashbuilder_flashcatalyst_workflows.html

Shneiderman, B. (2002). Promoting universal usability with multi-layer interface design. *Proceedings of the Acm sigcaph computers and the physically handicapped* (pp. 1-8). New York: ACM. doi: 10.1145/960201.957206

Visual identity and style guide. (2010). Informally published manuscript, Office of Public Relations, University of Northern Iowa, Cedar Falls, Iowa. Retrieved from http://www.uni.edu/ur/sites/default/files/UNI_Style_Guide.pdf

Resources

Andrews, P. (2010, July 21). 20 free web ui element kits and stencils. *Speckyboy Design Magazine*, Retrieved from http://speckyboy.com/2010/07/21/20-free-web-ui-element-kits-and-stencils/

Gube, J. (2009). 40+ helpful resources on user interface design patterns. *Smashing Magazine.* Retrieved from http://www.smashingmagazine.com/2009/06/15/40-helpful-resources-on-user-interface-design-patterns

Usability & user experience: an stc community - usability toolkit. (n.d.). Retrieved from http://www.stcsig.org/usability/resources/toolkit/toolkit.html#heuristics

Usability testing. (n.d.). *Usability.gov*, Retrieved from http://usability.gov/methods/test_refine/learnusa/index.html

Appendix A: Analysis of Existing Undergraduate Plan of Study

Welcome Screen

Pros	Cons
	• Has a LOT of text that a user will never read. (figure 1.1) • The user has to scroll down a lot to reach the main navigation into a plan of study. The navigation should be at the top of the plan where it is easily accessible. • Navigation Separates "Current Plan of Study" and "Hypothetical" Plans of study too much. The user can't easily switch between viewing a hypothetical plan to viewing their active plan. They must return to the welcome screen to go to the other. (figure 1.2) • Hypothetical plans are only stored for a small period of time. • A user cannot edit or manipulate hypothetical plan. Courses can only be added to active plans.

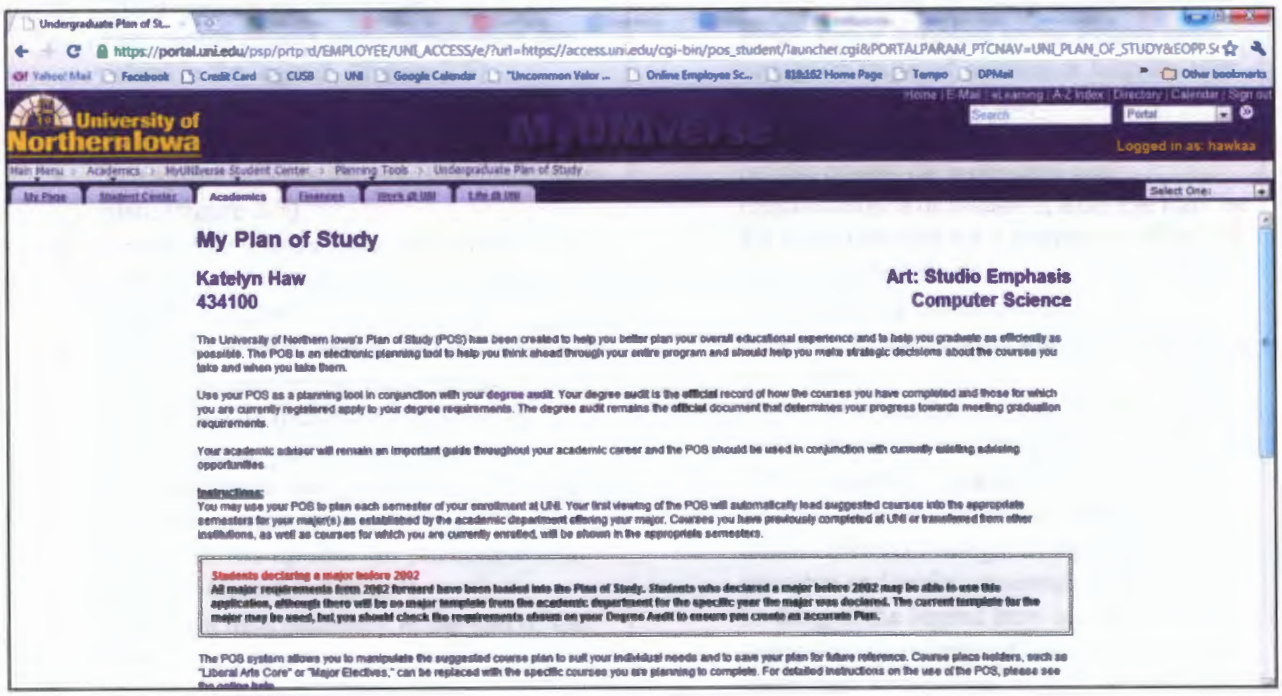


Figure 1.1. The top of the welcome screen

Appendix A: Analysis of Existing Undergraduate Plan of Study

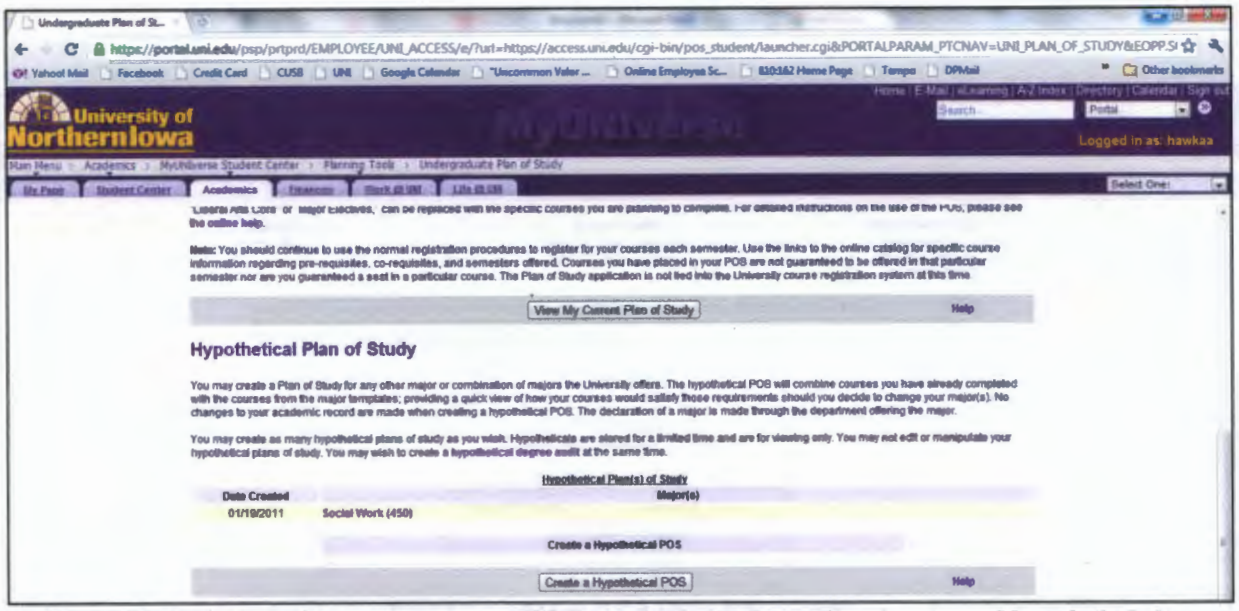


Figure 1.2. The bottom of the welcome screen with the navigation to the user's active plan and hypothetical plans

Active Plan of Study

Pros	Cons
• It color codes course depending on what requirement they fill. (figure 2.1 & 2.2) • The system allows you to customize the colors in the legend. (figure 2.5) • There are different print versions, including black and white. • The system allows the user to rebuild their plan. (figure 2.7) • The system allows the user to add a course by major, LAC course, or university elective. (figure 2.4) • There is a built in search feature to help the user to search for specific courses. • Requirements that could be filled by several classes are filled by course placeholders which give the user the option to choose a specific course to fill it. The system shows options for the course in a pop-up. (figure 2.8 & 2.9) • There is a nice summary at the end of the plan.	• The system has missing template several majors. Neither computer science or art studio major template so the system can't recognize required courses or help the user build a plan. (Figure 2.3) • What happens to the color coding when a course counts for more than one requirement? For instance, a course may be fill a requirement for a major as well as fill an LAC requirement. • Rebuilding a plan erases any customizations the user may have made to their plan. • The system has lots of pop-ups that aren't within the same window as the system. This is a poor UI feature. • Special courses are marked with codes to denote different things. Many codes are not intuitive and make no sense. Without looking at the legend how is a user supposed to understand what a "*" would mean next to a course? "<" ? "T"? • Clicking on a course number brings up the course catalog, not just the specified class. The user could get confused about which course was clicked.

Appendix A: Analysis of Existing Undergraduate Plan of Study

Figure 2.1. The top of the active plan of study

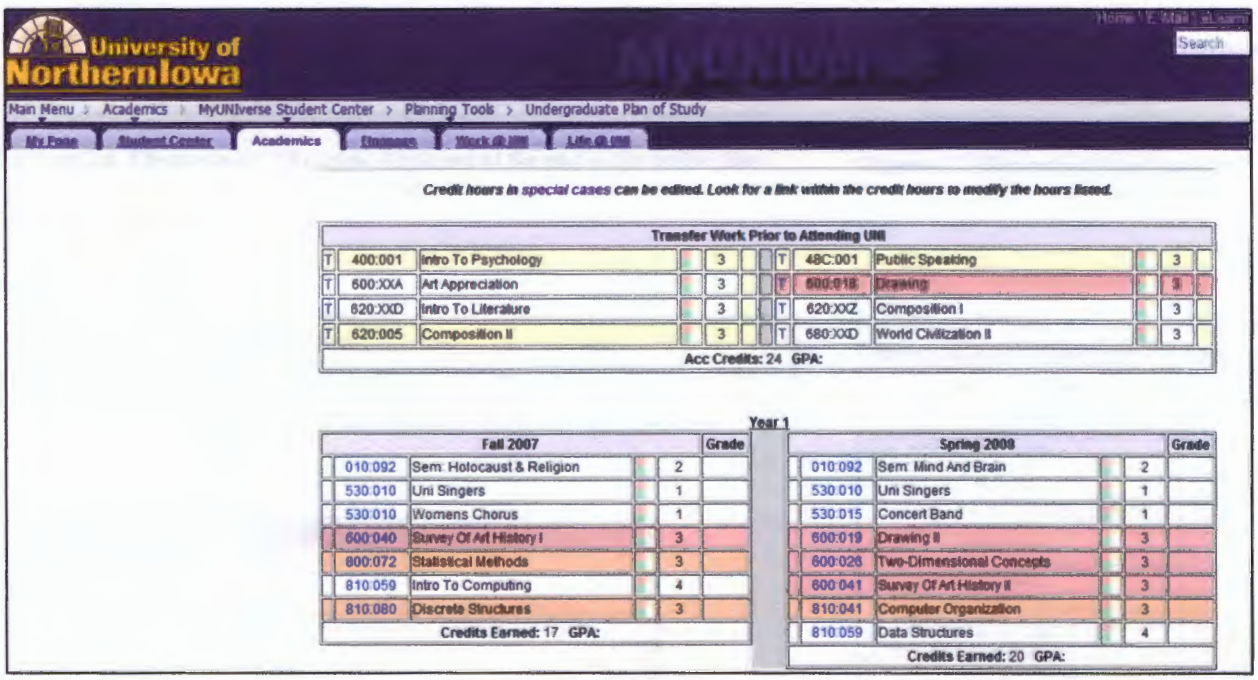


Figure 2.2. Sample plan of study

Appendix A: Analysis of Existing Undergraduate Plan of Study

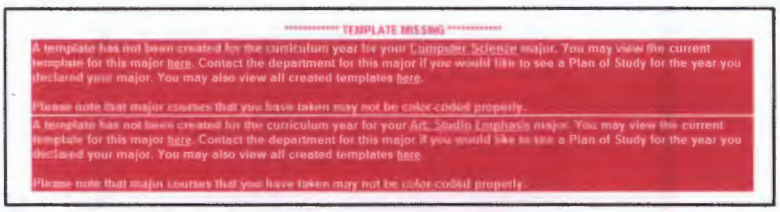


Figure 2.3. The system's error for missing templates

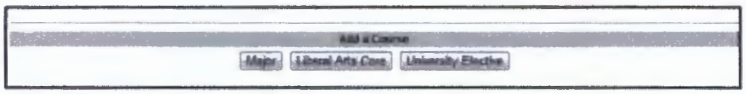


Figure 2.4. The user can add a course the plan from their major, the LAC core, or as a university elective

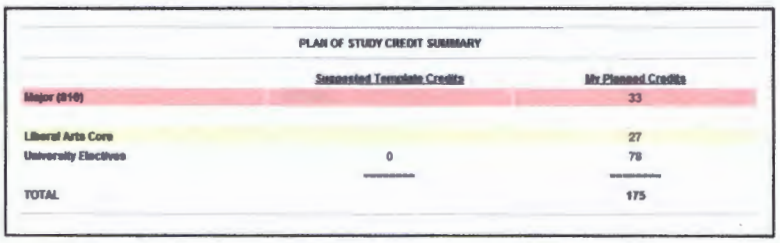


Figure 2.6. The summary of credits displayed at the end of the active plan of study

Figure 2.8. A sample LAC course holder from the plan of study

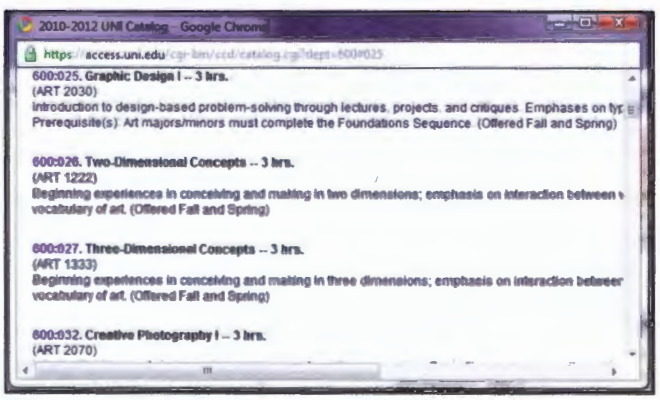


Figure 2.10. The system links to the course catalog to give the user descriptions and perquisites for courses

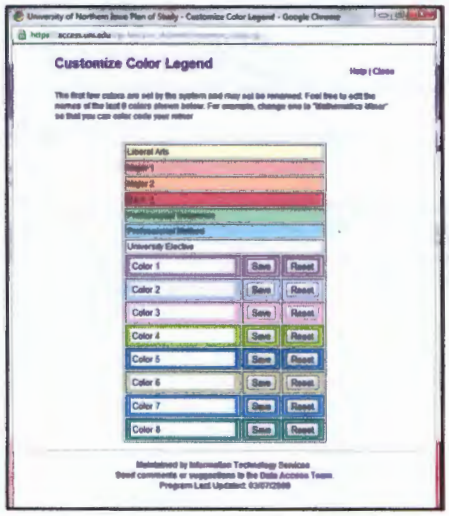


Figure 2.5. Editing the color legend

Figure 2.7. The rebuild plan function

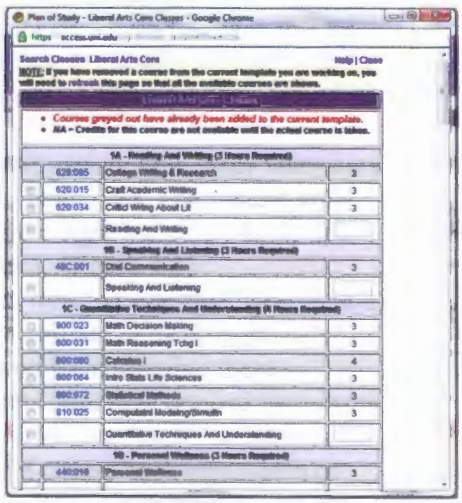


Figure 2.9. The LAC course listing to select a specific course to fulfill a LAC requirement

Appendix A: Analysis of Existing Undergraduate Plan of Study

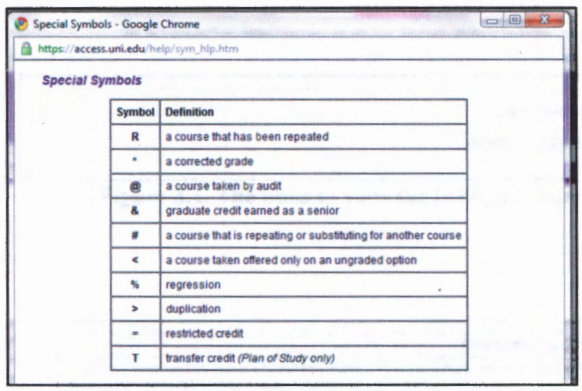


Figure 2.11 The legend for special symbols

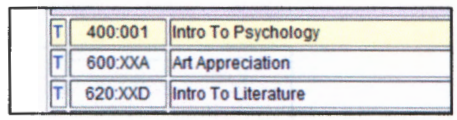


Figure 2.12-2.13. The codes to denote special types of courses

Creating a Hypothetical Plan of Study

Pros	Cons
• The system allows the user to view their major's program flow when in a hypothetical plan. • It is connect to the users' profile so it has information about previous classes the user has taken. • The system allows the user to choose what catalog they want to use for their degree.	• The system doesn't allow a user to add a minor or special program. • There is a maximum of 3 majors . • The user can't view their major's program flow in their active plan of study. • If a senior was adding a major as a junior of senior, the plan of study application is useless because it puts all courses that he or she hasn't taken but should have in a course holding area at the bottom without any indication of the order they should be taken in. • The user can't select how many semesters hey plan to be at UNI. The system only gives the user eight semesters to complete their degrees. (figure 3.2) • The system tells the user that "you should place these courses back into your Plan of Study. If you have satisfied these requirements, you may remove them from the Course Holding Area" but there is no way to put a course back into a plan of study or remove it from the holding area. (figure 3.2)

Appendix A: Analysis of Existing Undergraduate Plan of Study

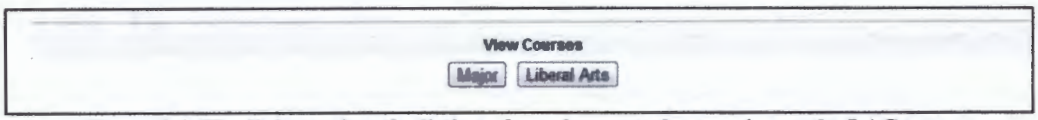


Figure 3.1. The links to view the listing of requirements for a major or the LAC core.

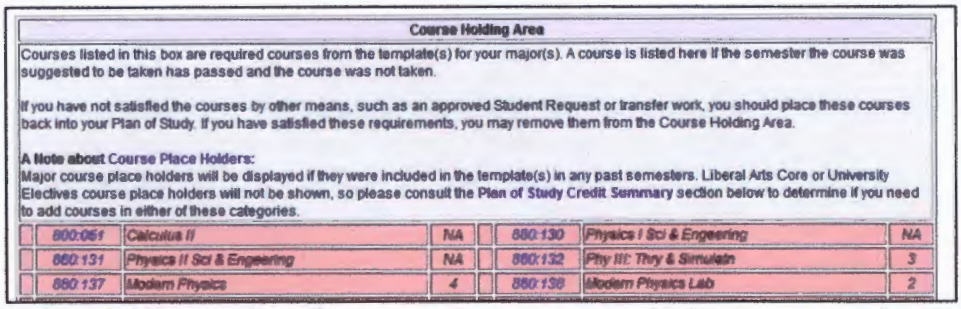


Figure 3.2. The course holding area that collects courses that were not completed in the semesters they should have been

Summary of Current System

The current system is incapable of meeting the needs of the majority of students at UNI. Its lack of flexibility and customizability make the tool difficult to use. The system doesn't allow the user to add more than 3 majors, minors, certificates, or any other special programs to their plan of study. A user can create multiple plans to compare majors but cannot switch a hypothetical plan to their active plan of study. Courses can't be moved or deleted from the schedule and most importantly it is practically useless for Juniors or Seniors looking add another major.

The user interface of the current system would is rudimentary and poorly organized. The system mixes tools to complete tasks in with the content of the plan. There is no clear menu or tool box structure to direct the user as to where to look for functionality. Pop-ups with more information show up in a different window instead of remaining inside the system. These pop-ups may not be supported well in every browser and could be blocked by anti-spam software. They also create an unnecessary

complication by enabling many different areas that the user must keep track of and jump between to accomplish a task. The system over informs the user by bombarding them with large paragraphs of text and instructions. This format limits the user's ability to scan through the data find the help they need.

Appendix B: Plan of Study Application UI Requirement Analysis

Task Analysis

- Freshman/Sophomore:
  A freshman/sophomore student would use the system to generate multiple hypothetical plans of study with different major combinations and compare them. A freshman/sophomore who is deciding on a major will be taking mostly LAC courses, and will not be as concerned about reviewing what program requirements they have completed and what they still have left as upperclassmen. Underclassmen who have already declared their major will focus on making sure they schedule for the appropriate courses to fill their program's requirements in the right time frame.

- Junior/Senior:
  Since most juniors and seniors have declared their majors, they would be less likely to use the hypothetical plan of study tool. Their use of the system would be primarily to determine which classes they should be registering for in the next semester. They would check to see what requirements they have completed and what they still had to complete.

- Transfer Student:
  The current system would be almost completely unhelpful for a junior or senior transfer student. The major worksheets are not flexible enough to account for a student entering into a program later than expected. A transfer student's need in a plan of study application would encompass both the needs of a freshman/sophomore looking for a major/majors to declare and a junior/senior student who is concerned with making sure they complete their program requirements.

- Advisors:
  A faculty advisor or an advisor from academic advising will assist students in making sure they meet program requirements in the right order and in the correct time frame. Their primary use of the system will be to assist students in creating and comparing hypothetical plans and reviewing current plans of study to determine which courses to schedule for in the upcoming semester. Their use of the system will be secondary to the students they advise.

Appendix C: Plan of Study Application UI Specification

System Name: Undergraduate Plan of Study

Purpose: Students are faced with overwhelming decisions when they begin their study at UNI. This system will help provide them with the information and recommendations that will help make important decisions easier. This system will aid students in creating a plan of study that encompasses all intended majors, minors, special programs, and certificates. It will help facilitate students' decision on what programs to take and enhance the interaction between advisees and advisers.

System Users:

- Students of the University of Northern Iowa
- Faculty and Staff that advise students on their plan of study

General Description: This application will be a web based application available to all UNI students from their MyUNIverse accounts. Students wishing to create a plan of study will enter the majors, minors, special programs, and certificates they wish to complete while at UNI. Taking into account the number of semesters the student wishes to remain at UNI, course prerequisites, and the semesters that courses are offered, the system will produce a personalize plan of study. The student would then have the ability to make changes to the proposed plan according to their personal preferences on semester course load, order of courses, etc. After each semester, the plan of study will be updated to reflect courses that have been completed.

Requirements (User's Tasks)

1.) Creating a Hypothetical Plan of Study

Hypothetical Plan of Study - is a proposed plan of study for a student's time here at UNI.

They system should create a *hypothetical plan of study* based on the following user inputs, user preferences, and system constraints.

Main Input

The user should be able to create a plan of study based on the following input:

- Majors, Minors, certificates, and special programs the user wishes to complete
- The course catalog they wish to follow for their major
- Priority levels for completion of majors, minors, certificates, and special programs
- The number of semesters they wish to spend at UNI

User Preferences

User Preferences – are constraints the user wishes their schedule should follow. They can be set or edited at an any time.

The user should be able to set the following preferences for their schedule.

- Maximum credit hours they wish to take in a single semester
- Minimum credit hours they wish to take in a single semester

Schedule Constraints:

Schedule Constraints – are policies and requirements set by the University. They cannot be broken without special permission to do so by professors, the registrar's office, or department heads.

The system should take into account the following constraints on a schedule when building a plan of study and should notify the user when their modifications to a plan of study exceed any constraint.

- Program Requirements
- Prerequisite Chains
- Previous Courses Taken/Transfer Credits
- Course Offerings (Every Semester, Fall, Spring, Variable)
- University Requirements for Maximum[1]/Minimum[2] credit hours per semester
- University Regulations for graduation requirements[3], Credit/No Credit[4], Audits, Student GPA, Transfer Credits, Electives, etc.

Output

The system should create a plan of study around their initial input and preferences as well the system constraints.

2.) Testing different Hypothetical Plans of Study

If a user is unsure what degrees or programs they wish to finish they can create multiple plans of study to compare. They system should allow them a way to test different combinations of majors, minors, etc. and create test plans of study. Test plans:

- The system should allow the user to test out different combinations of programs and save out different hypothetical plans
- The user should be able to create a new *hypothetical plan of study* from scratch
- The user should be able to create a new *hypothetical plan of study* from an existing *hypothetical plan of study* or an *active plan of study*
- The user should be able to access all saved hypothetical plans of study
- The user should be able to select one of the test plans as their active plan of study

3.) Setting a Hypothetical Plan of Study as the Active Plan of Study

[1] 18 credit Maximum without permission, 14 credit maximum for probationary students - http://www.uni.edu/catalog/academic-program-student-load

[2] 12 or more hours - http://www.uni.edu/catalog/classification-students

[3] Minimum of 120 hours, Minimum of 32 at UNI, Minimum 20 as a Jr/Sr,

[4] Not more than 22 hours of ungraded course work for undergraduates, Ungraded credit may not be applied to work required for a major, minor, or LAC except with the consent of the head of the department - http://www.uni.edu/catalog/creditno-credit-grading

Active Plan of Study - is the plan of study that the user has chosen as the plan they currently wish to follow.

- An *active plan of study* should remain fully editable and customizable
- After a *hypothetical plan of study* has been selected for to be the *active plan of study*, student should still be able to be able to create/modify hypothetical plans of study
- The system should allow users to switch their *active plan of study*

4.) Modifying a Hypothetical Plan of Study/Active Plan of Study

The user should be able to manual adjust and change a *hypothetical plan of study* or an *active plan of study* in the following ways: The system should notify the user when their modifications exceed their input or set preferences.

- Move courses from semester to semester
- Add another major, minor, certificate, or special program
- Select a specific emphasis area with their major or program if available
- Add elective courses they wish to take from a course catalog
- Delete university elective courses from the schedule (courses required for a major, minor, LAC, etc cannot be deleted entirely from a schedule)
- When a program has electives or multiple courses that meet a requirement, the user can select a specific course they wish to take to fill that requirement
- Select specific courses they wish to take to fill LAC requirements
- Add a summer term or winter break term
- Edit the number of semesters they wish to spend at UNI
- Edit their *user preferences*
- Add notes to a specific course
- Add notes to a specific semester
- Add global notes to their plan of study

5.) Adding/removing courses from the course holding area

Course holding area – an area within the student's plan of study they can place courses that they are planning on taking but do not know where they want to place them within the schedule.

The user should be able to complete the following tasks with the course holding area:

- Click and drag a course from within their schedule to the course holding area. (This includes required courses for their major, minor, etc, LAC courses, and university electives)
- Click and drag a course from the course holding area to a place within the schedule
- Add a new course directly to the course holding area
- Delete university elective courses from the course holding area (required courses cannot be deleted)

6.) Misc. Requirements for both hypothetical and active plans

- The user should be able to print off a plan

- The user should be able to add a global note to a plan

Requirements (System Feedback)

1. **Notifications when creating a new plan**
   The system should check for and inform the user of the following things when they are creating a new plan of study

   - They have more than one major, minor, certificate, etc within the same department
     (Some departments may not allow a person to receive more than one degree from the same department or there may be certain rules about course overlap)

2. **Plan Notifications**
   The system should check for, and inform the user when the following things occur within their plan of study

   High Level Notifications
   These notifications are considered of high importance and should be brought to the attention to the user for immediate consideration. A user can hide the notification but it cannot be dismissed entirely.

   - When their schedule exceeds other *schedule constraints*
   - The prerequisites have not been met for a course
   - When a major requires an application into a specific program
   - When their current schedule does not meet the requirements to fulfill all their desire programs or the University's graduation requirements

   Medium Level Notifications
   These notifications are considered of medium importance and should be brought to the attention of the user to deal with either immediately or to review later. A user can completely dismiss the notification.

   - When their schedule exceeds a *user preference*
   - When their schedule exceeds the set number of semesters at UNI
   - When upcoming semesters have unassigned courses (LAC courses, major electives, or university electives for which an exact course has been selected)

Requirements (System Information)

1. **Program Information**
   The system should allow the user to view the following information about any major, minor, or other program they include in their hypothetical/active plan

- The program requirements listed in the prerequisite order
- Program options such as emphasis or focus areas
- Any additional program requirements including:
    - GPA requirements
    - Program application requirements
    - Internship requirements
    - Undergraduate research requirements

2. **Semester Information**

   The following information should be provided for each semester
   - Semester (Fall, Spring, Summer) and year
   - Total credit hours in that semester
   - Semester GPA (if completed, NR if not)
   - Notes added to the semester by the user

3. **Course Information**

   The system should provide the following information about each course
   - Course number
   - Course title
   - Course description
   - Credit hours
   - Grade for the course (if already taken, NR if not)
   - Previously taught by (last 5 semesters or last 3 professors)
   - Offered: Every Semester, Spring, Fall, or Variable
   - Semester last offered
   - Next semester it is projected to be offered
   - Course tags:
       - Writing intensive
       - Lab course
       - Studio course
       - Graduate level course
       - Only available for credit/no credit
   - Notes added to the course by the user

Appendix D: User Tasks for Cognitive Walkthrough

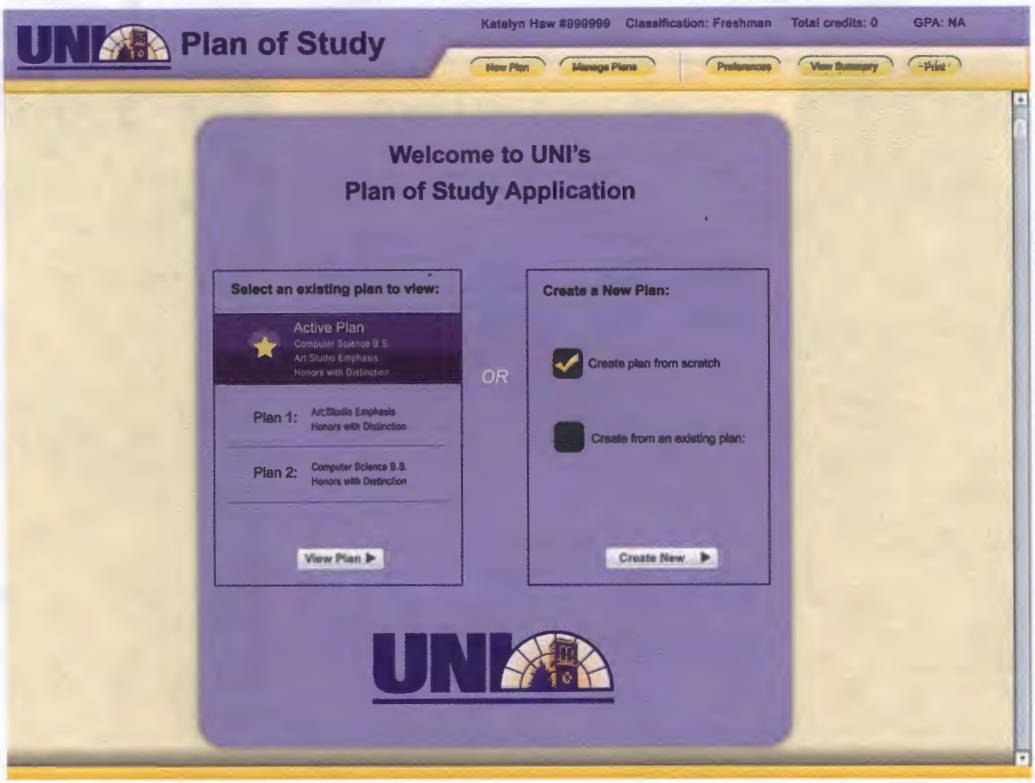
Hypothetical Plans

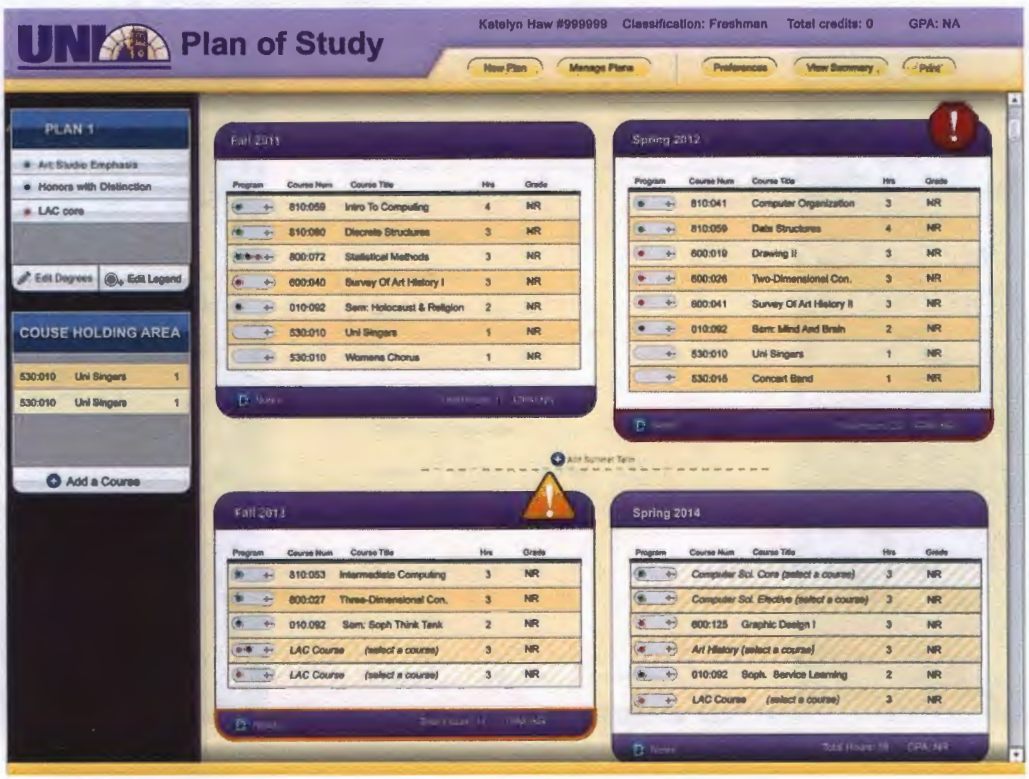
1. Create a Hypothetical Plan of study
    a. Select a Computer Science, B.A. major, 2010-2012, priority 1
    b. Select an Art Studio Emphasis major, 2010-2012, priority 2
    c. Select an honors with distinction special program, 2010-2012, priority 3
    d. Set the minimum number of credit hours you wish to take per semester at 16
    e. Set the maximum number of credit hours you wish to take per semester at 18
    f. Set the number of years you wish to spend at UNI to 4 years
2. Add a minor to your hypothetical plan
3. Change the priority of your minor to 3
4. Make another hypothetical plan of study
5. Set a hypothetical plan of study as your active plan of study

Active Plans

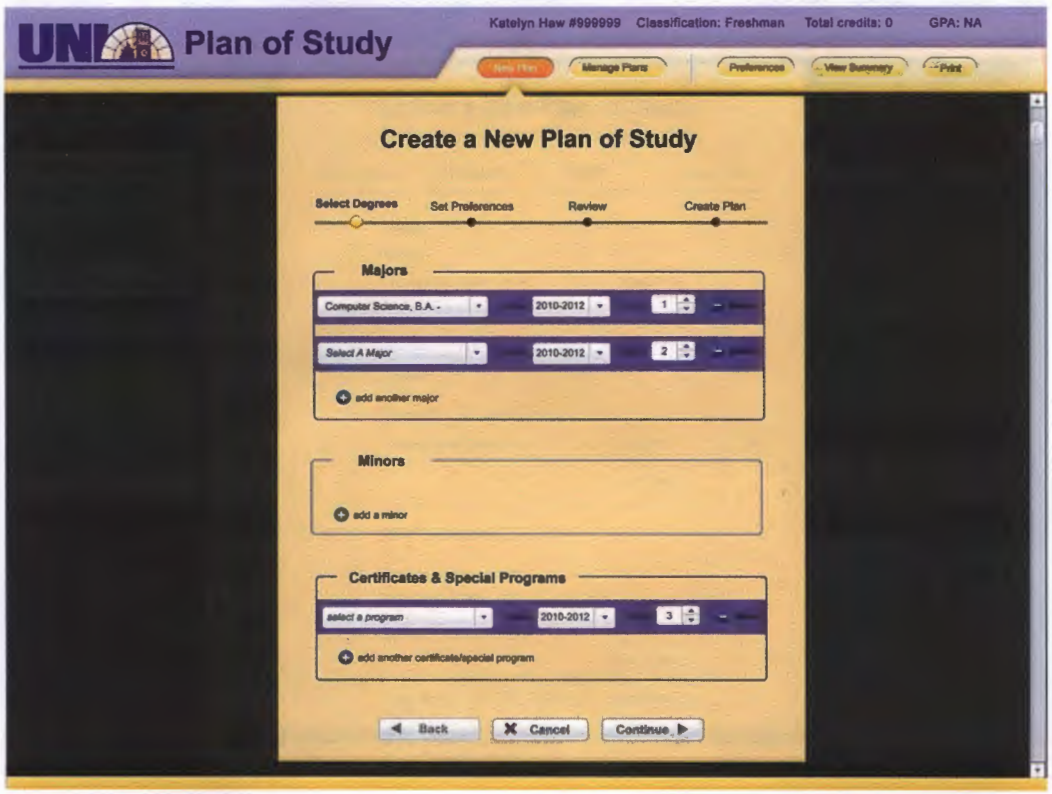
6. View your active plan of study
7. Explore all errors in the plan of study
8. Change the number of years at UNI
    a. Set the number of years you wish to spend at UNI to 4.5 years
9. Move a course to a different semester
10. Select a specific course for a LAC requirement
11. View the programs in this plan
12. Add a new color code for "courses I'm interested in"
13. Add a university elective course to the course holding area
14. Move a course from the holding area to the plan of study
15. Add the color code for "courses I'm interested in" to the university elective just added to the plan of study
16. View a course detail
    a. View the course description
    b. View who it was previously taught by
    c. View when it is offered
    d. View the last semester it was last offered and the next semester it is projected to be offered
    e. Determine if the course is writing intensive, a studio course, a lab course, a graduate level course or if it is only available for credit/no credit
    f. Add a note to the course
17. Add a note to a semester
18. View the plan summary
19. Print a copy of the active plan

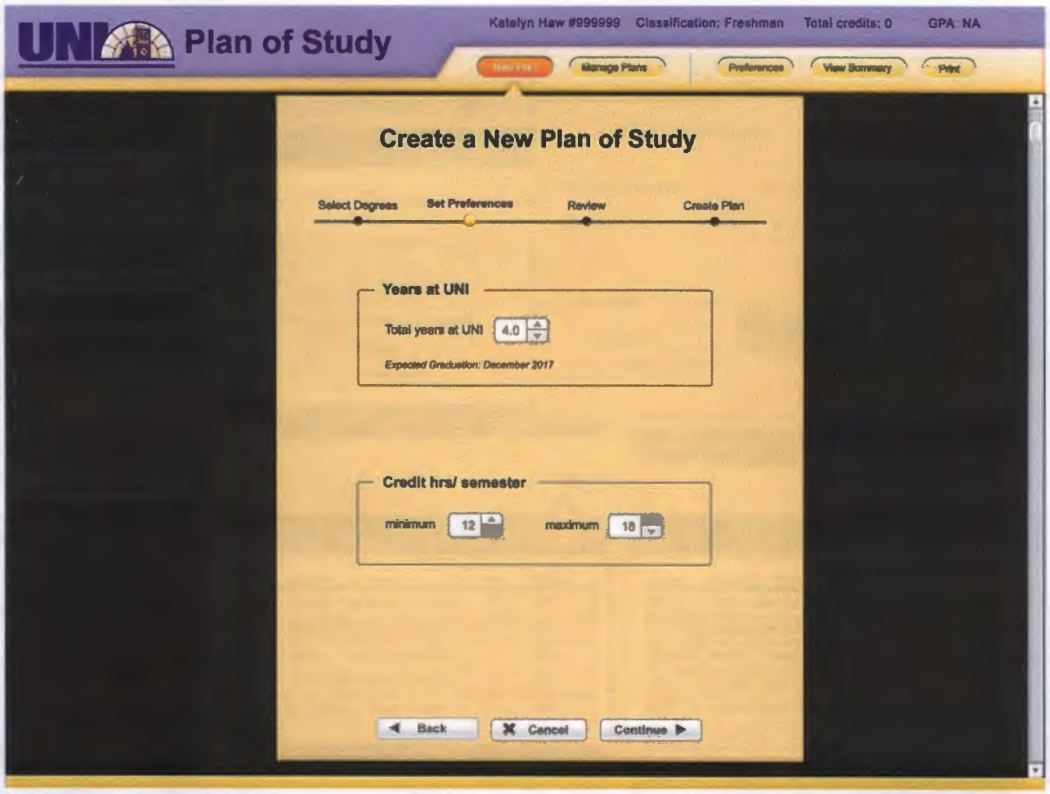
Appendix E: Screenshots of Final Color Wireframe



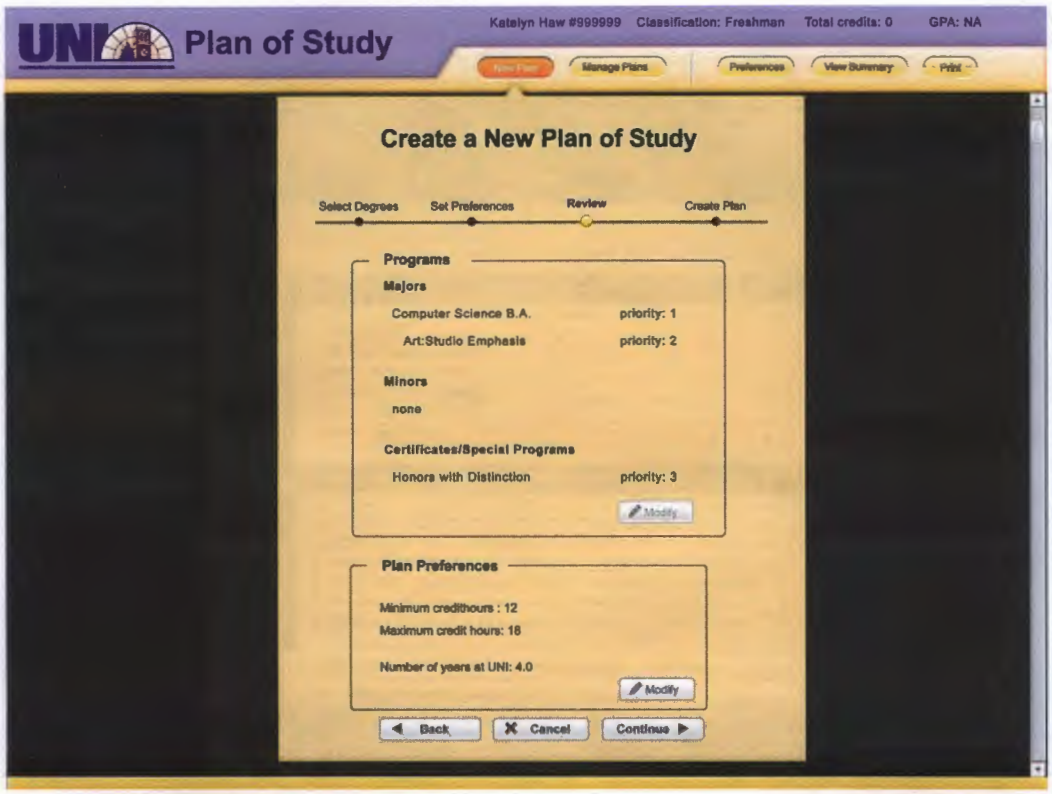


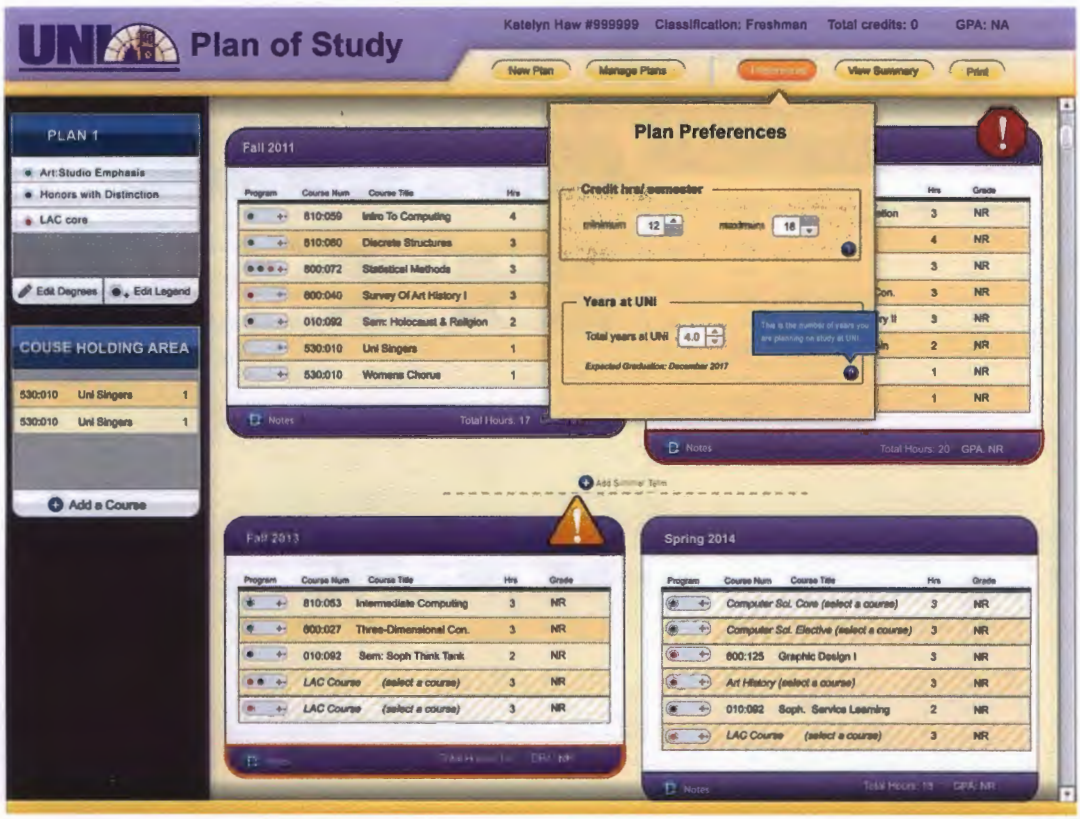
Appendix E: Screenshots of Final Color Wireframe



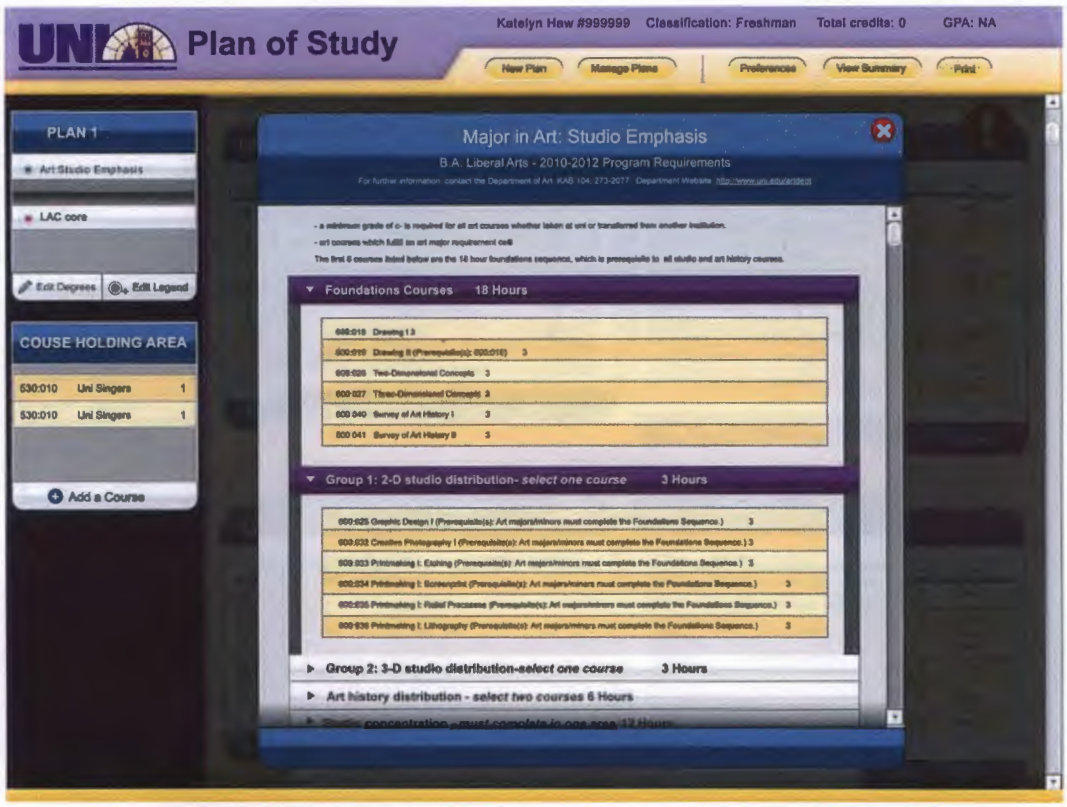


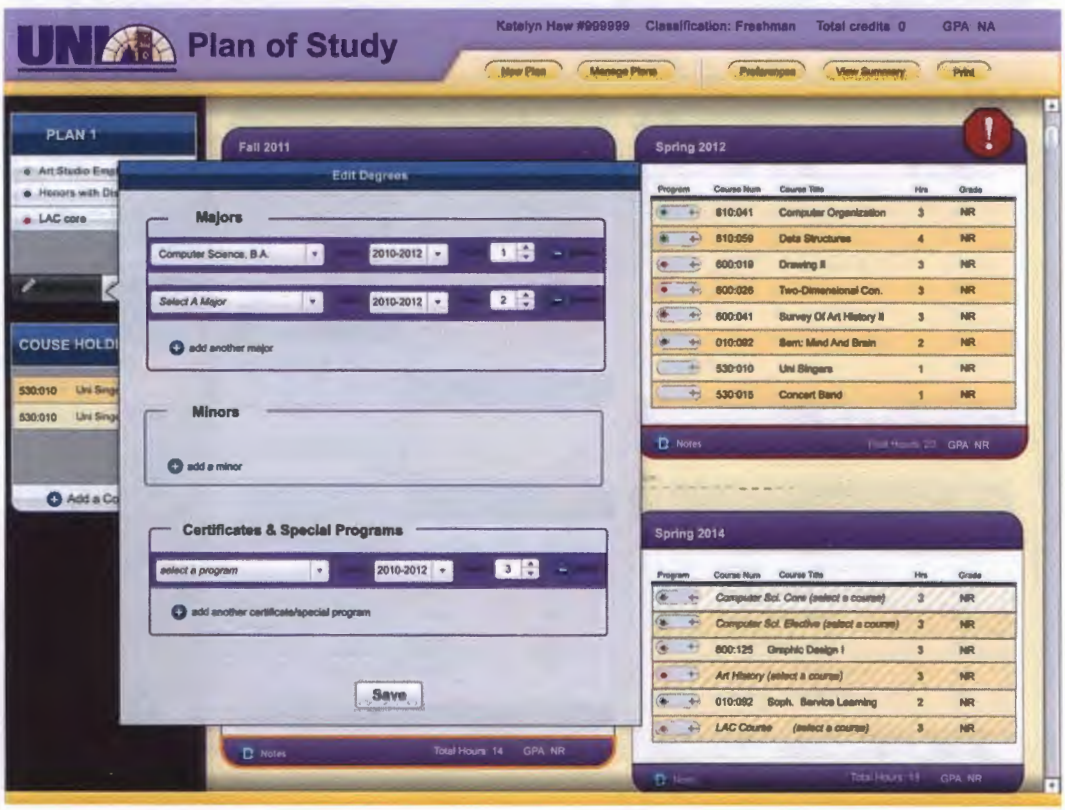
Appendix E: Screenshots of Final Color Wireframe



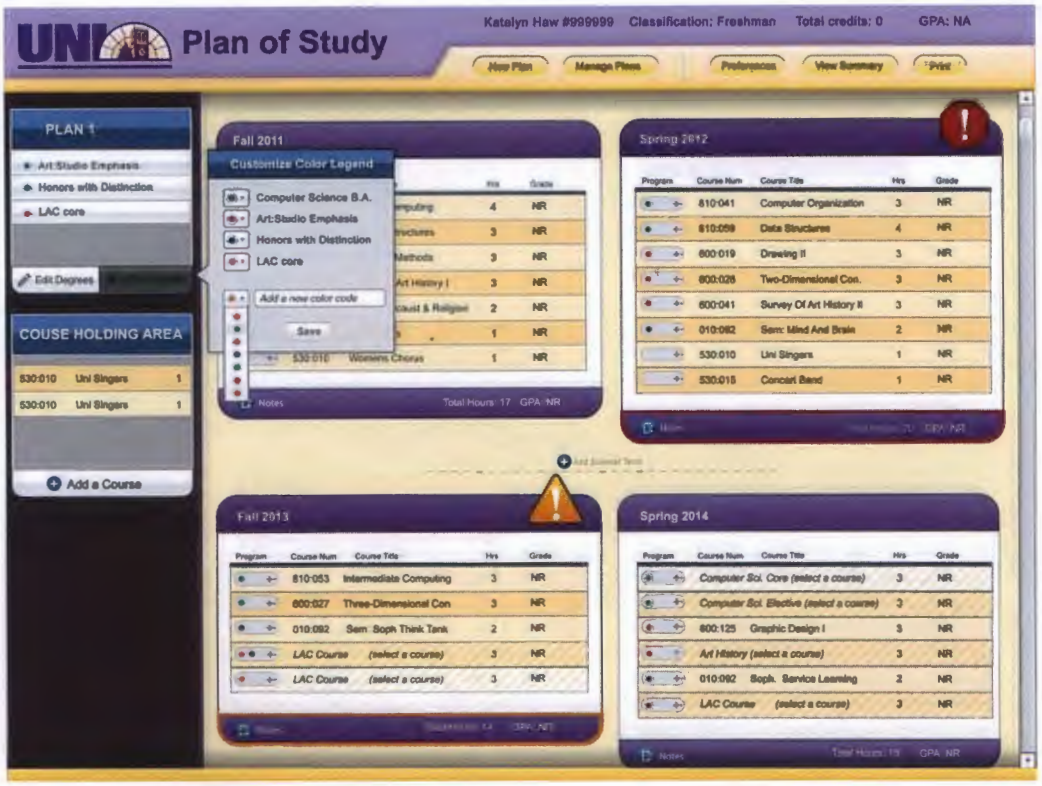


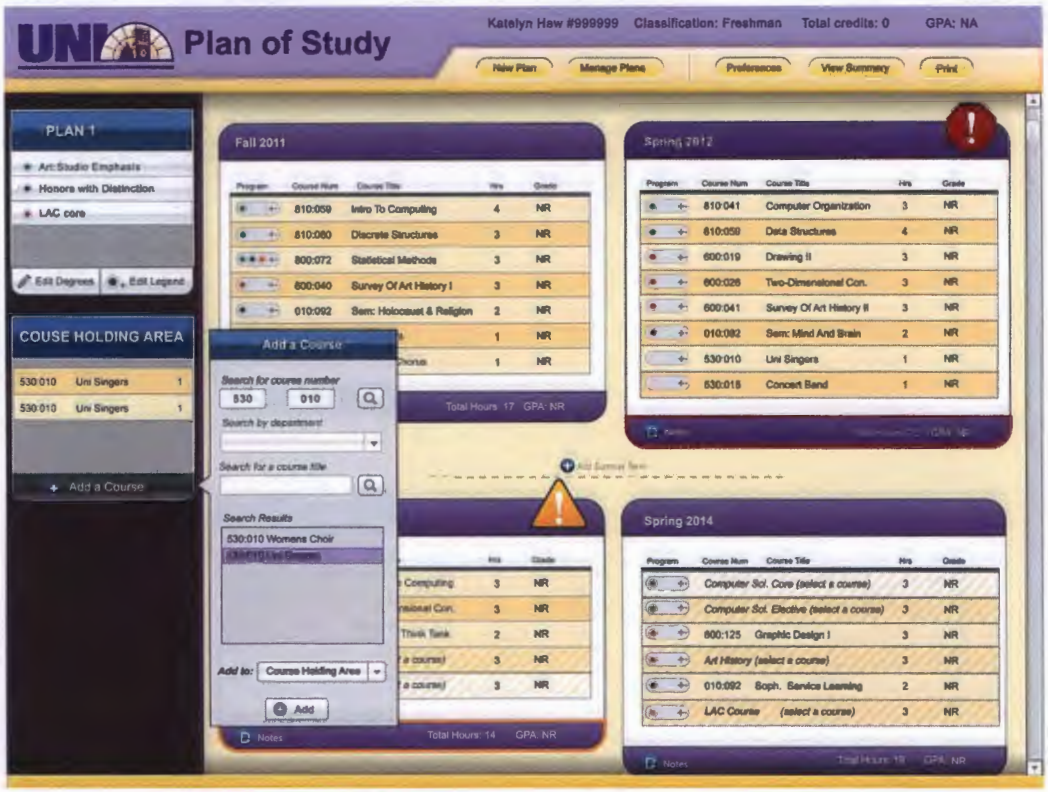
Appendix E: Screenshots of Final Color Wireframe



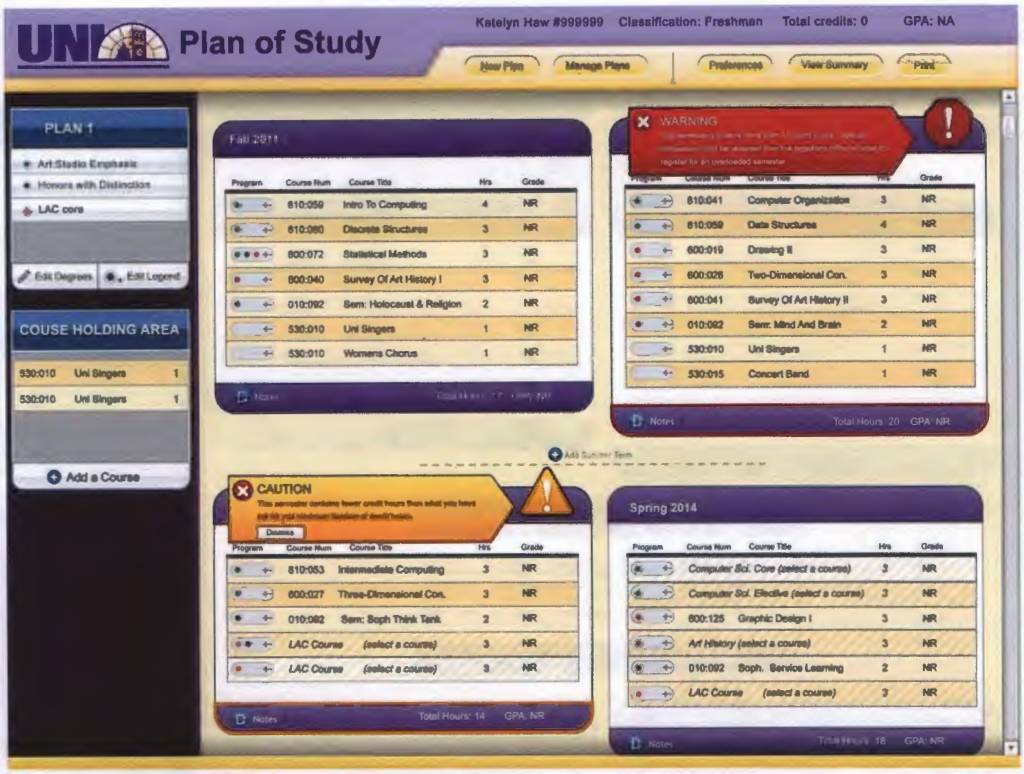


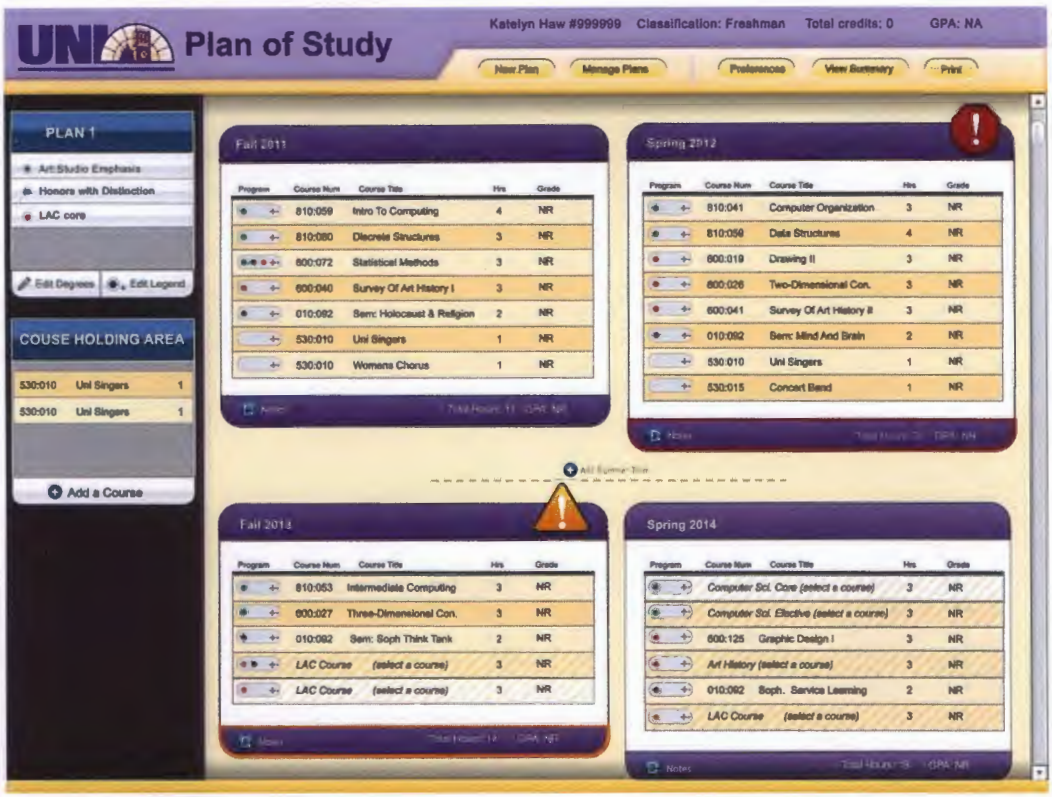
Appendix E: Screenshots of Final Color Wireframe



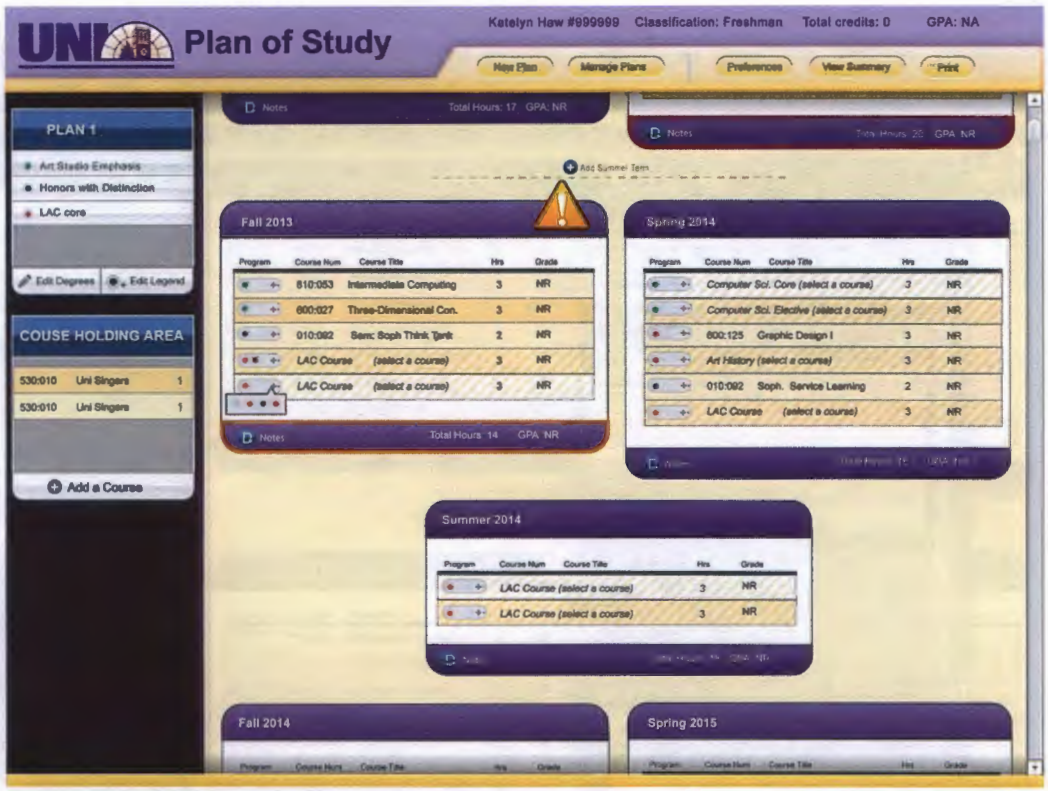


Appendix E: Screenshots of Final Color Wireframe

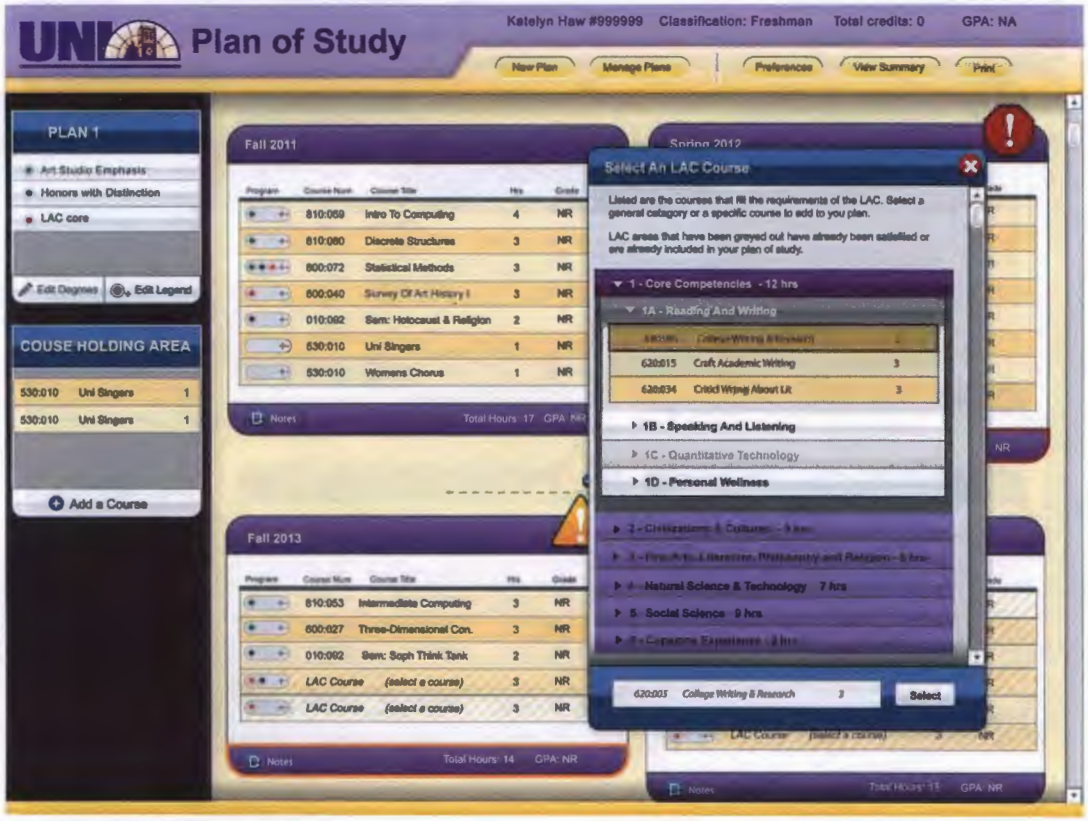


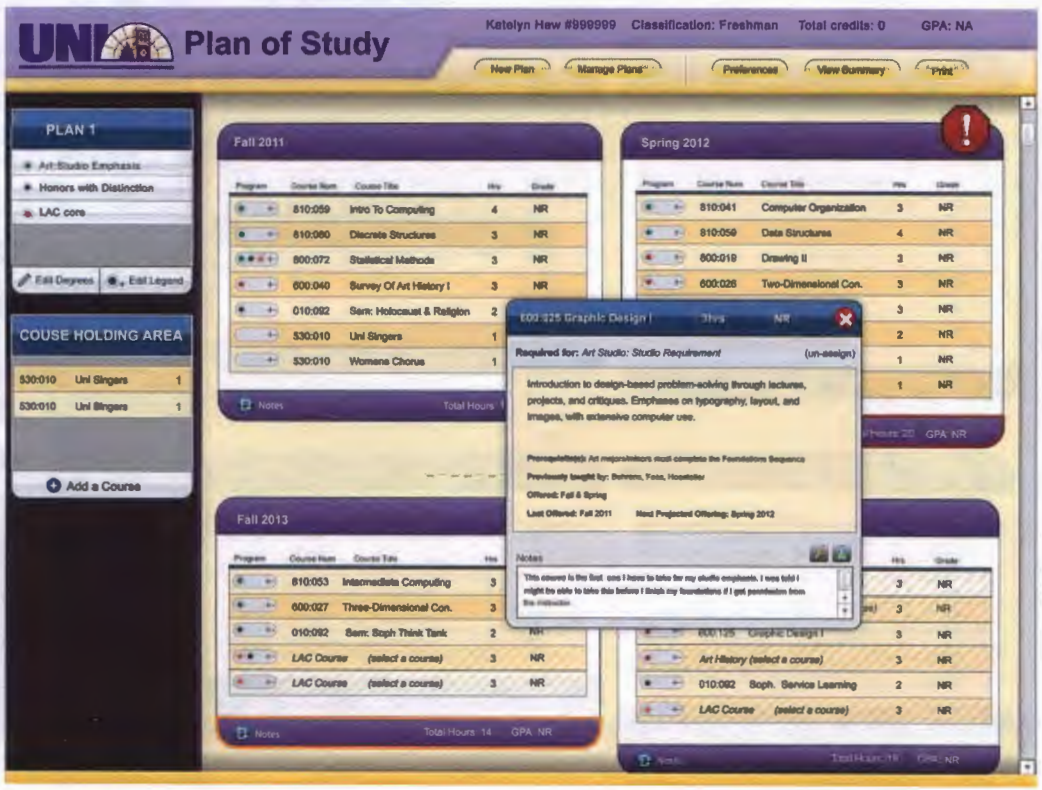


Appendix E: Screenshots of Final Color Wireframe

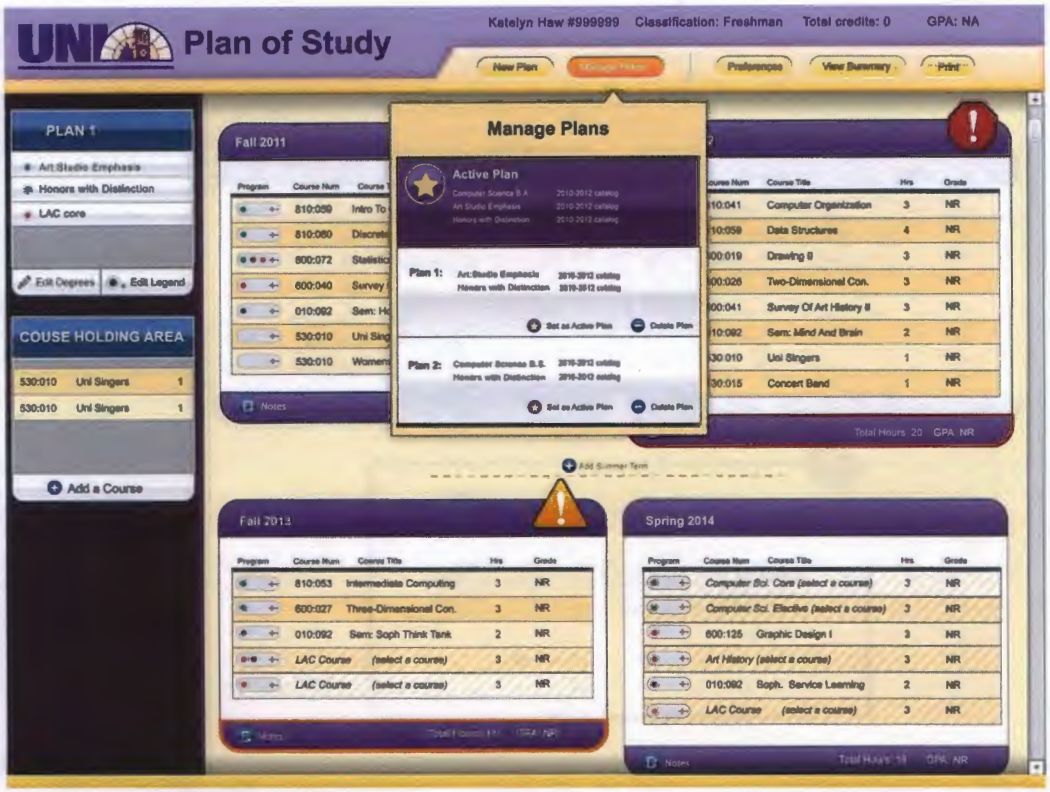


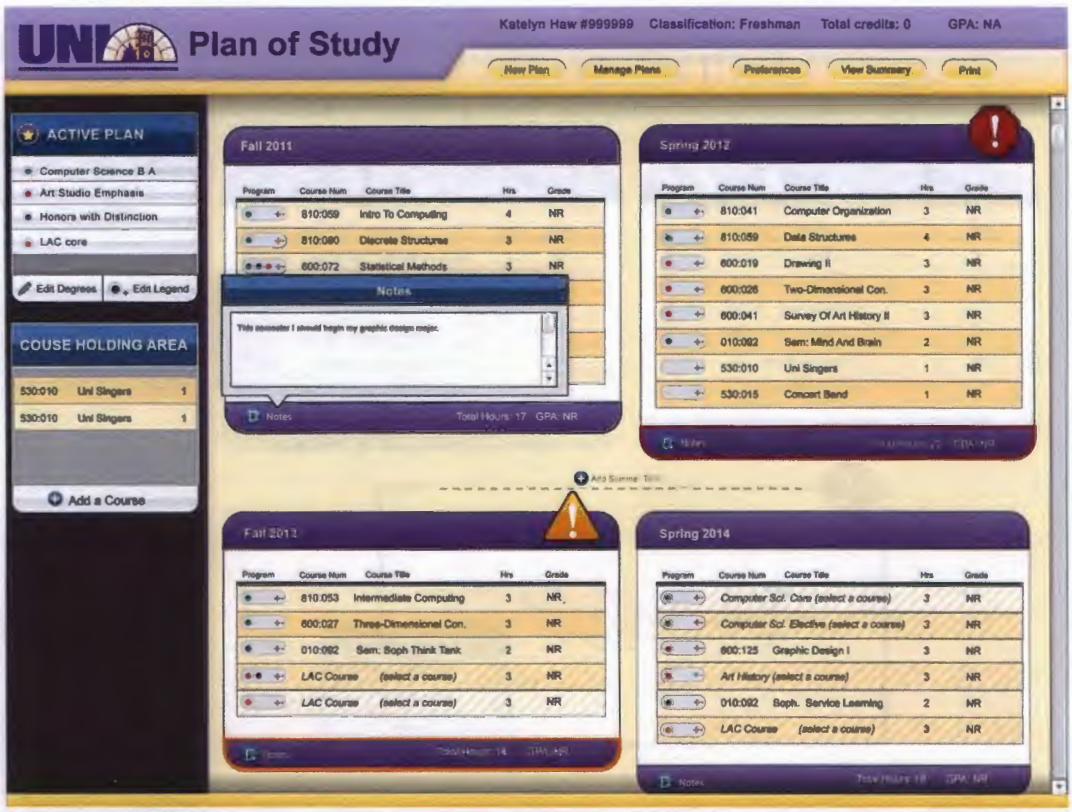
Appendix E: Screenshots of Final Color Wireframe



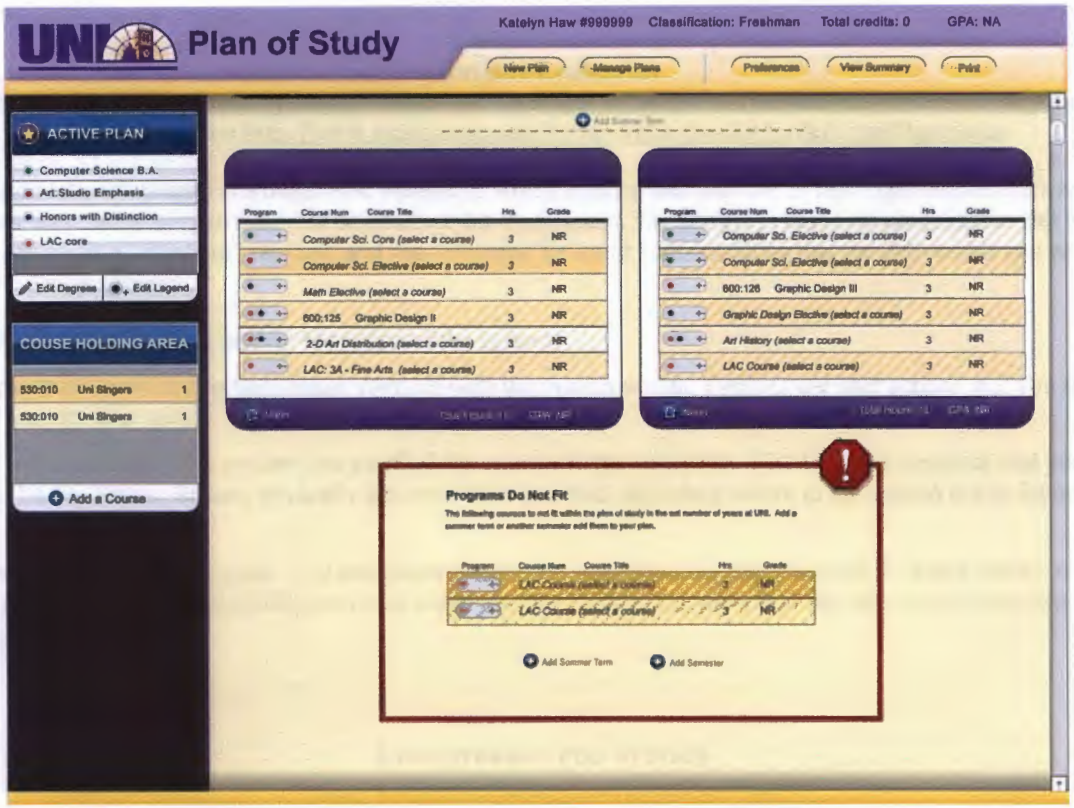


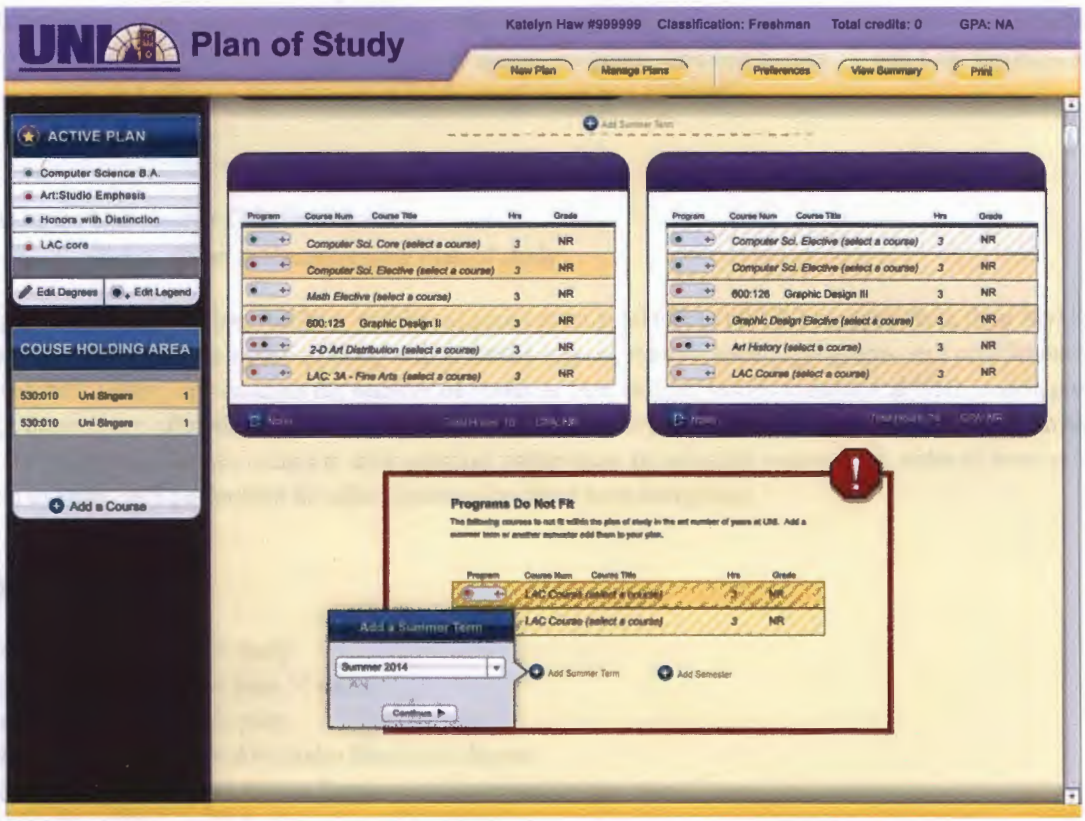
Appendix E: Screenshots of Final Color Wireframe





Appendix E: Screenshots of Final Color Wireframe





Appendix F: Heuristic Evaluation Results

Heuristic Evaluation

Based off of Ten Usability Heuristics by Jakob Nielsen and Usability & Technical Documentation, Xerox Corporation July 1995 modified from http://www.stcsig.org/usability/resources/toolkit/toolkit.html#heuristics

Heuristic Evaluation (expert review) is a diagnostic method in which experts take the role of less experienced users and describe the potential problems they see arising in a system or interface for those users. The review is based on compliance with a set of principles (heuristics). It is known as a "discount" method, and was designed for quick turnaround so that the deliverable can be attended to as part of an iterative design process.

1. Experiment with and establish a feel for the scope of the system.
2. Review the materials provided to familiarize yourself with the system design. Perform the user actions that you feel would be taken to perform the user tasks.
3. Identify and list any areas of the system that you feel are counter to the heuristics. List all of the concerns that you note, including what seem to be duplicates. Be sure to clearly describe what you find, including where in the system it was found.

Note: *This system is in a prototyping state. Not everything within the system will be fully functional. Many menus and options will be static. Evaluate the system to you best ability and note when you are unable to assume how the user would complete a task or how the system would respond.*

Undergraduate Plan of Study

http://www.uni.edu/hawkaa/thesis/Main.html

Purpose: Students are faced with overwhelming decisions when they begin their study at UNI. This system will help provide them with the information and recommendations that will help make important decisions easier. This system will aid students in creating a plan of study that encompasses all intended majors, minors, special programs, and certificates. It will help facilitate students' decision on what programs to take and enhance the interaction between advisees and advisers.

System Users:

- Students of the University of Northern Iowa
- Faculty and Staff that advise students on their plan of study

General Description: This application will be a web based application available to all UNI students from their MyUNIverse accounts. Students wishing to create a plan of study will enter the majors, minors, special programs, and certificates they wish to complete while at UNI. Taking into account the number of semesters the student wishes to remain at UNI, course prerequisites, and the semesters that courses are offered, the system will produce a personalize plan of study. The student would then have the ability to make changes to the proposed plan according to their personal preferences on semester course load, order of courses, etc. After each semester, the plan of study will be updated to reflect courses that have been completed.

Explore the following tasks:

1. View the active plan of study
2. Explore all errors in the plan of study
3. View the degrees in this plan
4. View the details for the Art:Studio Emphasis degree
5. Explore how you would Add or Remove a degree within this plan
6. Explore how you would Add a university elective course to the course holding area
7. Explore how you would Add a new color code to mark courses
8. Explore how you would add a color to a course in the plan of study

9. Add a note to a semester
10. Explore how you would select a specific course for a LAC course in the summer 2014
11. View the course details for *Graphic Design I*
    a. View the course description
    b. View who it was previously taught by
    c. View when it is offered
    d. View the last semester it was last offered and the next semester it is projected to be offered
    e. Add a note to the course

12. Explore how you would change the number of years at UNI
13. View the plan summary
14. Explore how you would view a different plan of study
15. Explore how you would change a different plan of study to the active plan of study
16. Explore how you would delete a plan of study

1. Visibility of System Status

The system should always keep user informed about what is going on, through appropriate feedback within reasonable time.

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
1.1	Does every display begin with a title or header that describes screen contents?	Yes				Yes	
1.2	Is there a consistent icon design scheme and stylistic treatment across the system?	N/A	Close	Yes		Yes	
1.3	Is a single, selected icon clearly visible when surrounded by unselected icons?	No	Add Color	No	Course colors	N/A	I'm not sure this is applicable
1.4	Do menu instructions, prompts, and error messages appear in the same place(s) on each menu?	Yes	Mostly aren't any (novice help)	Yes	When present. Save/Dismiss at bottom +/- in group	Yes	For the most part.
1.5	In multipage data entry screens, is each page labeled to show its relation to others?		Not sure what this refers to	Yes		Yes	
1.7	If pop-up windows are used to display error messages, do they allow the user to see the field in error?			N/A		Yes	The field in erro needs to be marked more clearly
1.8	Is there some form of system feedback for every operator action?	N/A	not any	No	limited prototype	N/A	This is hard to answer because the system is incomplete
1.9	After the user completes an action (or group of actions), does the feedback indicate that the next group of actions can be started?	No	Color, Color code, delete course, degree adds	N/A	Non modal - but windows act funny - prototype	Yes	

Appendix F: Heuristic Evaluation Results

1.1	Is there visual feedback in menus or dialog boxes about which choices are selectable?	N/A	Existence of dependent actions?	Yes	Generally clear	No	I had a hard time telling what was and what wasn't selectable.
1.11	Is there visual feedback in menus or dialog boxes about which choice the cursor is on now?	Yes		No	Selecting courses seems challenging	N/A	
1.12	If multiple options can be selected in a menu or dialog box, is there visual feedback about which options are already selected?	No	Mouse over to select courses	N/A	I think	Yes	
1.13	Is there visual feedback when objects are selected or moved?	N/A		Yes	Prototype	Yes	
1.14	Is the current status of an icon clearly indicated?		Not sure what you mean	N/A		Yes	
1.16	If there are observable delays (greater than fifteen seconds) in the system's response time, is the user kept informed of the system's progress?	N/A		N/A		N/A	
1.17	Are response times appropriate to the task?	N/A		Yes		Yes	
1.18	Typing, cursor motion, mouse selection: 50-1 50 milliseconds	N/A		Yes		N/A	
1.19	Simple, frequent tasks: less than 1 second	N/A		Yes		Yes	
1.2	Common tasks: 2-4 seconds	N/A		Yes		Yes	
1.21	Complex tasks: 8-12 seconds	N/A		Yes		N/A	
1.22	Are response times appropriate to the user's cognitive processing?	N/A		Yes		Yes	
1.23	Continuity of thinking is required and information must be remembered throughout several responses: less than two seconds.	N/A		Yes		Yes	
1.24	High levels of concentration aren't necessary and remembering information is not required: two to fifteen seconds.	N/A		Yes		Yes	
1.25	Is the menu-naming terminology consistent with the user's task domain?	No	Color code, course holding area, preferences	No	Degrees do not equal minors, programs?	Yes	
1.26	Does the system provide visibility: that is, by looking, can the user tell the state of the system and the alternatives for action?		Only with experience	Yes	Error Flags	No	This could be improved.
1.27	Do GUI menus make obvious which item has been selected?	Yes		No	LAC Course - prototype?	No	
1.28	Do GUI menus make obvious whether deselection is possible?	No		No	""	No	
1.29	If users must navigate between multiple screens, does the system use context labels, menu maps, and place markers as navigational aids?	No	?	Yes	Breadcrumbs on new plan - last element?	No	This could be improved. I felt lost in a few areas.

2. Match Between System and the Real World

The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
2.1	Are icons concrete and familiar?	Yes & No		Yes	Stop, Yeild, But color code is not clear	Yes	They could be visually improved.
2.2	Are menu choices ordered in the most logical way, given the user, the item names, and the task variables?	N/A		Yes		Yes	
2.3	If there is a natural sequence to menu choices, has it been used?	N/A		Yes	New, Edit, Context	Yes	For the most part.
2.4	Do related and interdependent fields appear on the same screen?	Yes & No		Yes	Add new course has some confusing on search criteria	Yes	
2.5	If shape is used as a visual cue, does it match cultural conventions?	No	Modify?	Yes	Stop, Yeild	Yes	
2.6	Do the selected colors correspond to common expectations about color codes?	N/A		Yes	Red, Yellow, to a degree purple	Yes	
2.7	When prompts imply a necessary action, are the words in the message consistent with that action?	No	The box at top left seems to be a label "Active Plan" and is confusing, should separate degree/major stuff & legend stuff	Yes		Yes	
2.9	On data entry screens, are tasks described in terminology familiar to users?	No		No	Degrees	Yes	
2.10	Are field-level prompts provided for data entry screens?	N/A	explicit prompts are avoided	Yes		N/A	
2.11	For question and answer interfaces, are questions stated in clear, simple language?	N/A		Yes		Yes	
2.12	Do menu choices fit logically into categories that have readily understood meanings?	N/A		Yes		Yes	
2.13	Are menu titles parallel grammatically?	N/A		No	Manage & New	No	"New Plan"/"Manage Plan" and "Preferences"/ "View

							Summary"/"Print " could be improved.
2.14	Does the command language employ user jargon and avoid computer jargon?	N/A		Yes		Yes	
2.15	Are command names specific rather than general?		?	Yes		Yes	
2.16	Does the command language allow both full names and abbreviations?		?	N/A		Yes	
2.17	Are input data codes meaningful?	N/A		N/A	Course Section - only place - match domain	Yes	
2.18	Have uncommon letter sequences been avoided whenever possible?	Yes		N/A		Yes	
2.20	Does the system automatically enter a dollar sign and decimal for monetary entries?	N/A		N/A		N/A	
2.22	Do GUI menus offer activation: that is, make obvious how to say *"now do it"?*	No		Yes	Add, Save	No	I was confused during a few tasks as to what I was supposed to do next.
2.23	Has the system been designed so that keys with similar names do not perform opposite (and potentially dangerous) actions?	N/A		N/A	Menus not keyed	N/A	

3. User Control and Freedom

Users should be free to select and sequence tasks (when appropriate), rather than having the system do this for them. Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Users should make their own decisions (with clear information) regarding the costs of exiting current work. The system should support undo and redo.

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y, N, N/ A	Comments
3.1	If setting up windows is a low-frequency task, is it particularly easy to remember?	N/A		Yes	Preferences	N/ A	
3.2	In systems that use overlapping windows, is it easy for users to rearrange windows on the screen?	No		No	Menus pop-up behind main pannel overlap/prototype? Stay on?	No	
3.3	In systems that use overlapping windows, is it easy for users to switch between windows?	No		No	Not part of real design I think	Yes	
3.4	When a user's task is complete, does the system wait for a signal from the user before processing?		?	No	Generally, Search by department?	N/ A	This is hard to answer because the system is incomplete.
3.6	Are users prompted to confirm commands that have drastic, destructive consequences?	N/A		Yes	Hopefully! - Delete Plan	No	
3.7	Is there an "undo" function at the level of a single action, a data entry, and a complete group of actions?	N/A		No	But easily reversible	No	
3.8	Can users cancel out of operations in progress?	No		Yes		N/ A	
3.9	Are character edits allowed in commands?	N/A		N/A	?	N/ A	I don't know.
3.10	Can users reduce data entry time by copying and modifying existing data?	N/A		N/A	Could not edit course – Prototype	N/ A	I don't know.

3.11	Are character edits allowed in data entry fields?	N/A		No		N/A	
3.12	If menu lists are long (more than seven items), can users select an item either by moving the cursor or by typing a mnemonic code?	N/A		No	Not apparent yet	Yes	
3.13	If the system uses a pointing device, do users have the option of either clicking on menu items or using a keyboard shortcut?	No		Yes		Yes	
3.14	Are menus broad (many items on a menu) rather than deep (many menu levels)?	N/A		N/A		Yes	
3.15	If the system has multiple menu levels, is there a mechanism that allows users to go back to previous menus?	N/A		N/A		Yes	
3.16	If users can go back to a previous menu, can they change their earlier menu choice?	N/A		N/A		Yes	
3.17	Can users move forward and backward between fields or dialog box options?	N/A		N/A		Yes	
3.18	If the system has multipage data entry screens, can users move backward and forward among all the pages in the set?	Yes		N/A		Yes	
3.19	If the system uses a question and answer interface, can users go back to previous questions or skip forward to later questions?	N/A		N/A		N/A	
3.20	Do function keys that can cause serious consequences have an undo feature?	N/A		N/A		No	
3.21	Can users easily reverse their actions?	No	Not known but no obvious opportunity	Yes		Yes	
3.22	If the system allows users to reverse their actions, is there a retracing mechanism to allow for multiple undos?	N/A		No	Not apparent	No	
3.23	Can users set their own system, session, file, and screen defaults?	No	e.g. zoom	N/A	Preference - not quite same	No	This could be helpful.

Appendix F: Heuristic Evaluation Results

4. Consistency and Standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y, N, N/ A	Comments
4.1	Have industry or company formatting standards been followed consistently in all screens within a system?	N/A		No	UNI Standards from Bus. Office logo colors.	Yes	
4.2	Has a heavy use of all uppercase letters on a screen been avoided?	Yes		No	Labels for L nav	Yes	
4.3	Do abbreviations not include punctuation?	N/A		Yes		Yes	
4.4	Are integers right-justified and real numbers decimal-aligned?	N/A		No	Bad tab stop for hours in major	Yes	
4.5	Are icons labeled?	Yes		No		Yes	
4.6	Are there no more than twelve to twenty icon types?		Not sure	Yes		Yes	
4.7	Are there salient visual cues to identify the active window?	Yes	Only 1!	No	Overlap - no difference in active	Yes	
4.8	Does each window have a title?	Yes	Only 1	No	no title on main page	Yes	
4.9	Are vertical and horizontal scrolling possible in each window?	Yes	Yes, But dual and unneeded browser scrolls	No	vertical scrolling nasty	No	
4.10	Does the menu structure match the task structure?	No	"Manage Plans" & Preferences" have unclear connotations	Yes		Yes	
4.11	Have industry or company standards been established for menu design, and are they applied consistently on all menu screens in the system?	N/A		N/A		Yes	

4.12	Are menu choice lists presented vertically?	N/A		Yes		Yes	
4.13	If "exit" is a menu choice, does it always appear at the bottom of the list?	N/A		N/A		Yes	
4.14	Are menu titles either centered or left-justified?	N/A		Yes		Yes	
4.15	Are menu items left-justified, with the item number or mnemonic preceding the name?	N/A		N/A	L.J. no item #	Yes	
4.16	Do embedded field-level prompts appear to the right of the field label?	Yes	but not always clear	Yes	or Above	Yes	
4.17	Do on-line instructions appear in a consistent location across screens?	N/A	no instructions available	Yes	Hover above ?	Yes	
4.18	Are field labels and fields distinguished typographically?	N/A		Yes	By control type	Yes	This could be improved.
4.19	Are field labels consistent from one data entry screen to another?	N/A		Yes		Yes	
4.20	Are fields and labels left-justified for alpha lists and right-justified for numeric lists?	N/A		No	Some problems in semester course list	Yes	
4.21	Do field labels appear to the left of single fields and above list fields?	N/A		No		Yes	
4.22	Are attention-getting techniques used with care?	N/A		Yes		Yes	For the most part. I think the errors in the plan of study are a bit too large.
4.23	Intensity: two levels only	N/A		N/A		Yes	
4.24	Size: up to four sizes	N/A		Yes		No	
4.25	Font: up to three	N/A		Yes		No	The fonts used in the errors in the plan of study are too small. They are very hard to read.
4.26	Blink: two to four hertz	N/A		N/A		N/A	
4.27	Color: up to four (additional colors for occasional use only)	Yes		Yes	Not contrast for outline - icon is only visible cue	Yes	
4.28	Sound: soft tones for regular positive feedback, harsh for rare critical conditions	N/A		N/A		N/A	
4.29	Are attention-getting techniques used only for exceptional conditions or for	Yes		Yes		Yes	

	time-dependent information?						
4.30	Are there no more than four to seven colors, and are they far apart along the visible spectrum?	Yes	Think so	No	Red/Orange in Legend not distinct	Yes	
4.31	Is a legend provided if color codes are numerous or not obvious in meaning?	Yes		Yes		Yes	
4.32	Have pairings of high-chroma, spectrally extreme colors been avoided?	Yes	I think so	Yes		Yes	
4.33	Are saturated blues avoided for text or other small, thin line symbols?	Yes		Yes		Yes	
4.34	Is the most important information placed at the beginning of the prompt?	N/A		Yes		Yes	
4.35	Are user actions named consistently across all prompts in the system?	No	Edit vs Modify	Yes	Aside from "New" Plan	Yes	
4.36	Are system objects named consistently across all prompts in the system?	Yes		Yes		Yes	
4.37	Do field-level prompts provide more information than a restatement of the field name?	N/A		No		Yes	
4.38	For question and answer interfaces, are the valid inputs for a question listed?	N/A		N/A		Yes	
4.39	Are menu choice names consistent, both within each menu and across the system, in grammatical style and terminology?	N/A		No	New Plan	Yes	
4.40	Does the structure of menu choice names match their corresponding menu titles?	N/A		Yes		Yes	
4.41	Are commands used the same way, and do they mean the same thing, in all parts of the system?	N/A		Yes	Prototype	Yes	
4.42	Does the command language have a consistent, natural, and mnemonic syntax?	N/A		N/A		Yes	
4.43	Do abbreviations follow a simple primary rule and, if necessary, a simple secondary rule for abbreviations that otherwise would be duplicates?	N/A		N/A	yes GPA	Yes	
4.44	Is the secondary rule used only when necessary?	N/A		N/A		Yes	
4.45	Are abbreviated words all the same length?	N/A		N/A		Yes	
4.46	Is the structure of a data entry value consistent from screen to	N/A		Yes		Yes	

	screen?						
4.48	If the system has multipage data entry screens, do all pages have the same title?	N/A		N/A		Yes	
4.49	If the system has multipage data entry screens, does each page have a sequential page number?	N/A		N/A		No	This would be helpful on the "Create a New Plan of Study" screens.
4.51	Are high-value, high-chroma colors used to attract attention?	No			Red is muted - low value	No	

5. Help Users Recognize, Diagnose, and Recover From Errors

Error messages should be expressed in plain language (NO CODES).

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
5.1	Is sound used to signal an error?	No		No		No	
5.2	Are prompts stated constructively, without overt or implied criticism of the user?	N/A		N/A		Yes	
5.3	Do prompts imply that the user is in control?	N/A		Yes		Yes	
5.4	Are prompts brief and unambiguous.	N/A		Yes		Yes	
5.5	Are error messages worded so that the system, not the user, takes the blame?	N/A		Yes	To the extent reasonable	Yes	For the most part. I think the error message wording could be improved.
5.6	If humorous error messages are used, are they appropriate and inoffensive to the user population?	N/A		N/A		N/A	
5.7	Are error messages grammatically correct?	N/A		No	Set for "you" minimum - registrars	Yes	
5.8	Do error messages avoid the use of exclamation points?	N/A		Yes	Outside of Icon	Yes	
5.9	Do error messages avoid the use of violent or hostile words?	N/A		Yes		Yes	
5.10	Do error messages avoid an anthropomorphic tone?	N/A		Yes		YEs	
5.11	Do all error messages in the system use consistent grammatical style, form,	N/A		Yes		Yes	

	terminology, and abbreviations?						
5.12	Do messages place users in control of the system?	N/A		Yes		Yes	
5.13	Does the command language use normal action-object syntax?	N/A		N/A		Yes	
5.14	Does the command language avoid arbitrary, non-English use of punctuation, except for symbols that users already know?	N/A		N/A		Yes	
5.15	If an error is detected in a data entry field, does the system place the cursor in that field or highlight the error?	N/A			Prototype	Yes	
5.16	Do error messages inform the user of the error's severity?	N/A		Yes	Presumably	Yes	For the most part.
5.17	Do error messages suggest the cause of the problem?	N/A		Yes		Yes	
5.18	Do error messages provide appropriate semantic information?	N/A		Yes		Yes	
5.19	Do error messages provide appropriate syntactic information?	N/A		Yes		Yes	
5.20	Do error messages indicate what action the user needs to take to correct the error?	N/A		Yes	caution implied	Yes	

6. Error Prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

#	**Review Checklist**	**Evaluator 1**		**Evaluator 2**		**Evaluator 3**	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
6.2	Have dots or underscores been used to indicate field length?	N/A		No		No	
6.3	Is the menu choice name on a higher-level menu used as the menu title of the lower-level menu?	N/A		No		Yes	
6.4	Are menu choices logical, distinctive, and mutually exclusive?	Yes		Yes		Yes	
6.6	If the system displays multiple windows, is navigation between windows simple and visible?	N/A		No	Menu windows overlap	Yes	
6.11	Does the system prevent users from making errors whenever possible?	Yes		N/A	Prototype	No	This could be improved.
6.12	Does the system warn users if they are about to make a potentially serious	No		N/A	Prototype	N/A	

	error?						
6.13	Does the system intelligently interpret variations in user commands?			No		No	
6.14	Do data entry screens and dialog boxes indicate the number of character spaces available in a field?	No	Do notes have a max length?	No		Yes	
6.15	Do fields in data entry screens and dialog boxes contain default values when appropriate?	N/A		No	Prototype	Yes	

7. Recognition Rather Than Recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

#	**Review Checklist**	**Evaluator 1**		**Evaluator 2**		**Evaluator 3**	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
7.1	For question and answer interfaces, are visual cues and white space used to distinguish questions, prompts, instructions, and user input?	N/A		N/A		Yes	
7.2	Does the data display start in the upper-left corner of the screen?		Not sure	Yes		Yes	
7.3	Are multiword field labels placed horizontally (not stacked vertically)?	N/A		Yes		Yes	
7.4	Are all data a user needs on display at each step in a transaction sequence?	Yes		Yes		Yes	I think so.
7.5	Are prompts, cues, and messages placed where the eye is likely to be looking on the screen?	N/A		Yes		No	The location of the errors in the plan of study could be improved.
7.6	Have prompts been formatted using white space, justification, and visual cues for easy scanning?	N/A		Yes		No	
7.7	Do text areas have "breathing space" around them?	N/A		Yes		No	This could be improved.
7.8	Is there an obvious visual distinction made between "choose one" menu and "choose many" menus?	N/A		No	LAC Course - Prototype	No	
7.9	Have spatial relationships between soft function keys (on-screen cues) and keyboard function keys been	N/A		N/A		N/A	

	preserved?						
7.10	Does the system gray out or delete labels of currently inactive soft function keys?	N/A		Yes		N/A	
7.11	Is white space used to create symmetry and lead the eye in the appropriate direction?	N/A		Yes		Yes	
7.12	Have items been grouped into logical zones, and have headings been used to distinguish between zones?	No		Yes		Yes	
7.13	Are zones no more than twelve to fourteen characters wide and six to seven lines high?	Yes		No	Course List - by semester	Yes	
7.14	Have zones been separated by spaces, lines, color, letters, bold titles, rules lines, or shaded areas?	Yes		Yes		Yes	
7.15	Are field labels close to fields, but separated by at least one space?	N/A		Yes		Yes	
7.16	Are long columnar fields broken up into groups of five, separated by a blank line?	N/A		No		Yes	
7.17	Are optional data entry fields clearly marked?	No		No		Yes	
7.18	Are symbols used to break long input strings into "chunks"?	N/A		No		Yes	
7.19	Is reverse video or color highlighting used to get the user's attention?	N/A		Yes		N/A	
7.20	Is reverse video used to indicate that an item has been selected?		Not Sure	Yes		N/A	
7.21	Are size, boldface, underlining, color, shading, or typography used to show relative quantity or importance of different screen items?	No		Yes		Yes	
7.22	Are borders used to identify meaningful groups?	N/A		Yes		Yes	
7.23	Has the same color been used to group related elements?	No		Yes		Yes	
7.24	Is color coding consistent throughout the system?	Yes		Yes		Yes	
7.25	Is color used in conjunction with some other redundant cue?	Yes	A little	No		Yes	
7.26	Is there good color and brightness contrast between image and background colors?	Yes	I think so	Yes		Yes	
7.27	Have light, bright, saturated colors been used to emphasize data and have darker, duller, and desaturated colors been used to de-emphasize data?	Yes		No & N/A		Yes	
7.28	Is the first word of each menu choice the most important?	N/A		No	?	Yes	
7.29	Does the system provide mapping: that is, are the relationships between controls and actions apparent to the user?	No		No	Program column no idea how to operate	N/A	

7.30	Are input data codes distinctive?	N/A		yes		Yes	
7.31	Have frequently confused data pairs been eliminated whenever possible?	N/A		N/A		Yes	
7.32	Have large strings of numbers or letters been broken into chunks?	N/A		Yes	810:072	Yes	
7.33	Are inactive menu items grayed out or omitted?	N/A		Yes		Yes	
7.34	Are there menu selection defaults?	N/A		Yes	Default for majors both = 1	Yes	
7.35	If the system has many menu levels or complex menu levels, do users have access to an on-line spatial menu map?	N/A		N/A		No	
7.36	Do GUI menus offer affordance: that is, make obvious where selection is possible?	Yes		Yes	std controls	No	
7.37	Are there salient visual cues to identify the active window?	N/A		N/A		Yes	
7.39	Do data entry screens and dialog boxes indicate when fields are optional?	N/A		N/A		No	
7.40	On data entry screens and dialog boxes, are dependent fields displayed only when necessary?	N/A		N/A		No	

8. Fexibility and Minimalist Design

Accelerators-unseen by the novice user-may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions. Provide alternative means of access and operation for users who differ from the "average" user (e.g., physical or cognitive ability, culture, language, etc.)

#	**Review Checklist**	**Evaluator 1**		**Evaluator 2**		**Evaluator 3**	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
8.3	Can users define their own synonyms for commands?	N/A		No		No	
8.4	Does the system allow novice users to enter the simplest, most common form of each command, and allow expert users to add parameters?	N/A		No		No	
8.9	If menu lists are short (seven items or fewer), can users select an item by moving the cursor?	N/A		No		Yes	
8.10	If the system uses a type-ahead strategy, do the menu items have mnemonic codes?	N/A		N/A		N/A	I don't think so.
8.11	If the system uses a pointing device, do users have the option of either clicking on fields or using a keyboard shortcut?	No		No		N/A	

9. Aesthetic and Minimalist Design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y, N, N/A	Comments	Y, N, N/A	Comments	Y,N, N/A	Comments
9.1	Is only (and all) information essential to decision making displayed on the screen?	No	Only but not all	Yes		No	
9.2	Are all icons in a set visually and conceptually distinct?		Not sure	Yes		Yes	
9.3	Have large objects, bold lines, and simple areas been used to distinguish icons?	No		Yes		Yes	
9.4	Does each icon stand out from its background?	Yes		Yes		Yes	I think the icons could be enhanced.
9.6	Are meaningful groups of items separated by white space?	Yes		Yes		Yes	
9.7	Does each data entry screen have a short, simple, clear, distinctive title?	No		Yes		Yes	
9.8	Are field labels brief, familiar, and descriptive?	N/A		Yes		Yes	
9.9	Are prompts expressed in the affirmative, and do they use the active voice?	N/A		Yes & N/A	Add to course holding area	Yes	
9.10	Is each lower-level menu choice associated with only one higher level menu?	N/A		No & N/A	Course holding area add course ? I suspect so	Yes	
9.11	Are menu titles brief, yet long enough to communicate?	N/A		Yes		Yes	
9.12	Are there pop-up or pull-down menus within data entry fields that have many, but well-defined, entry options?		? Not sure	Yes		Yes	

10. Help and Documentation

Appendix F: Heuristic Evaluation Results

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
10.4	If menu choices are ambiguous, does the system provide additional explanatory information when an item is selected?	No		No		No	
10.5	Are data entry screens and dialog boxes supported by navigation and completion instructions?	No		Yes		Yes	
10.6	If menu items are ambiguous, does the system provide additional explanatory information when an item is selected?	No		Yes		No	
10.7	Are there memory aids for commands, either through on-line quick reference or prompting?	No	but not needed	No		No	
10.8	Is the help function visible; for example, a key labeled HELP or a special menu?	No		Yes		No	
10.9	Is the help system interface (navigation, presentation, and conversation) consistent with the navigation, presentation, and conversation interfaces of the application it supports?	Yes		Yes		N/A	
10.10	Navigation: Is information easy to find?	No			Based on limited prototype Y or N/A	Yes	
10.11	Presentation: Is the visual layout well designed?	Yes	Mostly			Yes	I think a stronger hierarchy of information could be established.
10.12	Conversation: Is the information accurate, complete, and understandable?	N/A				Yes	
10.13	Is the information relevant?		Ok for experienced user but not the novice			Yes	
10.14	Goal-oriented (What can I do with this program?)	No				Yes	
10.15	Descriptive (What is this thing for?)	No				Yes	
10.16	Procedural (How do I do this task?)	No				No	
10.17	Interpretive (Why did that happen?)	No				Yes	

10.18	Navigational (Where am I?)	Yes				Yes	
10.19	Is there context-sensitive help?	No		Yes		Yes	
10.20	Can the user change the level of detail available?	No		No		No	
10.21	Can users easily switch between help and their work?	N/A		Yes		N/A	
10.22	Is it easy to access and return from the help system?	N/A		Yes		N/A	
10.23	Can users resume work where they left off after accessing help?	N/A		Yes		N/A	

11. Skills

The system should support, extend, supplement, or enhance the user's skills, background knowledge, and expertise ----not replace them.

#	**Review Checklist**	**Evaluator 1**		**Evaluator 2**		**Evaluator 3**	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
11.1	Can users choose between iconic and text display of information?	No		N/A		No	
11.2	Are window operations easy to learn and use?		Not sure	Y		Yes	
11.3	If users are experts, usage is frequent, or the system has a slow response time, are there fewer screens (more information per screen)?		not able to tell (no backend)	N/A		Yes	
11.4	If users are novices, usage is infrequent, or the system has a fast response time, are there more screens (less information per screen)?	No		N/A		Yes	
11.5	Does the system automatically color-code items, with little or no user effort?	N/A			?Don't fully understand	Yes	
11.7	Are users the initiators of actions rather than the responders?	Yes		Yes		Yes	
11.22	Does the system correctly anticipate and prompt for the user's probable next activity?	N/A		No		Yes	This is hard to answer because the system is incomplete.

12. Pleasurable and Respectful Interaction with the User

The user's interactions with the system should enhance the quality of her or his work-life. The user should be treated with respect. The design should be aesthetically pleasing- with artistic as well as functional value.

#	Review Checklist	Evaluator 1		Evaluator 2		Evaluator 3	
		Y,N, N/A	Comments	Y,N, N/A	Comments	Y,N, N/A	Comments
12.1	Is each individual icon a harmonious member of a family of icons?		Not sure	Yes		Yes	
12.2	Has excessive detail in icon design been avoided?	Yes		Yes		Yes	
12.3	Has color been used with discretion?	Yes		Yes		Yes	
12.4	Has the amount of required window housekeeping been kept to a minimum?	Yes		Yes		Yes	
12.6	Has color been used specifically to draw attention, communicate organization, indicate status changes, and establish relationships?	No		Yes	Faint Borders on Warnings	Yes	
12.7	Can users turn off automatic color coding if necessary?	N/A		N/A		No	
12.14	If the system supports graphical tasks, has an alternative pointing device been provided?	No		No		N/A	

Additional Evaluator Comments

Evaluator 1:

- Page should fit window (at least not have scroll bars that are not needed)
- Internal scroll bar is confusing and touchy
- Can't zoom (I have poor computer vision)
- On opening page, tabs don't work (purple thing in front)
- Warning and caution not legible for me
- Warning pop-up vs caution popup vs non-pop up warning - Why 3?
- *"View the details for the Art: Studio Emphasis degree"* - pop-up that cannot be moved aside for dual viewing
- Not clear how to add/remove a degree from plan - No add for majors, no select/delete to remove
- Not sure how to add a note to a semester
- Don't know how to select a specific course for an LAC course
- Only pieces of courses active to select and view course details
- No close/cancel when adding a course or a note
- No back button
- Mouse-over of courses is not over all the box
- No university electives category
- Create new plan should always start at beginning
- Consider different layout or color coding or … for semesters – hard to distinguish them
- Don't understand how to "create" a new plan of study
- Preference Menu – name not clear

Appendix F: Heuristic Evaluation Results

Evaluator 2:

- Initial Page – Plan 1, Plan 2 – cut off text
- Buttons @ top row have names behind splash – not able to see
- Diff between active plan vs alt. plans
- Difference between ! errors and warnings
    - Color suggest severity
    - Clicking confirms
- Scrolling is a challenge
- Intrigued by course holding area – programs do not fit courses added here?
- Legend difficult to work with
- Same sense of important double dip courses
- Some are not coded – by student interest?
- Unclear on hashing means – prerequisites needed?
- Blank entries
- Total house don't add up in Fall 11
- + on legend suggests something should happen – but not obvious / functional
- 9.13 – Font size – too small
- View degrees – not functional for CS BA but for art studio
- "Studio Emphasis" – Degree Name? - Unclear if major until pop-up vs program like honors or LAC
- White on white for selected text
- Expand Columns not aligned
- Small font
- Art major - Difficult to understand via prototype - assume dropdowns are different requirements
- Add/Remove degree
    - No corresponding + for degree
    - Unsure @ dropdown next to major
    - Minors / certificates = degrees
- Add a course
    - Search icon lets me look up #
    - Dept no search trigger b selection – loss focus?
    - Scrollable?
    - Dept. does not show up in list
- Add new color code
    - One at a time instead of add button
    - Triggered by edit legend instead of add code
    - How does add new color code match to degree components – by defaults what else does this mean?
    - "Fun"
    - Removal of degree comps means what here
- Unclear how to add a color to a course in the plan of study – unclear why I am doing this
- Can't see where to select a specific course for an LAC course in summer 2014
- View the course details for Graphic Design I? – couldn't complete task
- Do I have to set a plan as active
- What does set as active plan mean?