James Wen

Positive Motion, Inc.

james.wen@positivemotion.com

Abstract

The simplicity of flashcards makes them ideally suited for memorizing word lists on small, mobile devices but flashcards are not well suited to subjects that require more sophisticated organizational structure. Legal outlines, for example, do not lend themselves well to the simple associative lists of flashcards. We describe how we adapted and enhanced user-interface design techniques so that they may be more effectively applied to the design of small, mobile devices in an unproven space rather than the more standard desktop applications for which a large body of user data exists.

Introduction

While mobile phones are ideally suited as replacements for cumbersome flashcards, a substantial body of content that students have to commit to memory do not lend themselves well to the simple format of flashcards. For example, a law student has to commit to memory, procedural options that define the steps in the prosecution of a lawsuit. Such advanced concepts are often organized as outlines where general steps are divided into specific substeps which themselves may be further divided in sub-substeps and so on. With a tremendous amount of content to master, law students are motivated to find study tools to help in the memorization process, including mobile tools. However, unlike flashcards—which suggest a very obvious user-interface given their simple front-back association—hierarchical outlines may invite numerous interface possibilities and it is not immediately obvious which would be best suited as a memorization—as opposed to a browsing—interface.

In our search for a solution, we found that, while numerous design guidelines for small, mobile applications have been proposed, the design process itself has had less exposure in the same domain. In this paper, we share some of the adaptations we made to a standard user-interface design process so that it would be more appropriate for designing mLearning applications. Two challenges that exist in the mLearning domain are the constraints imposed by the physical limitations and the dearth of usage data that could be leveraged as part of a standard design process. We also introduce a visualization technique we found useful in the development process and we illustrate an application of the design methodology in a case study that examines our solution.

Traditional and Mobile User-Interface Design Techniques

Numerous user-interface design methodologies exist, ranging from the classic top-down waterfall model to the highly iterative star life cycle to the reusable design pattern approach. Because most user-interface design methodologies are geared towards sophisticated desktop applications such as business programs, games or websites, we chose to use the Logical User Centered Interactive Design (LUCID) method, which focuses more on the user experience and is less dependent on the scale of the application. The phases of the LUCID approach are:

1. Envision the goals, requirements, constraints and functionalities
2. Analyze the user's mental model and habits
3. Design the prototype
4. Evaluate usability and design detailed interaction elements
5. Build and test

We made some slight adjustments to the phases because we found that, while this method is generally platform independent, limitations of small, mobile devices imposed such severe

constraints, that it is difficult to envision functionalities without better understanding the users. Because the field of mobile applications is relatively new, there is not a wealth of data on existing habits for use in design purposes. From a user-acceptance perspective, then, solutions for small, mobile devices are far more speculative in nature than their desktop counterparts. We therefore found it more effective to envision relevant constraints and functionalities after some user behavior data is acquired. While this may blur the distinct between product design and interface design, we believe that the perspective is not so much that product design occurs later in the process as much as interface design occurs earlier.

We also split the analyze phase into two distinct stages in order to clearly delineate the transition from user requirements to finding a solution in the constrained environment of small, mobile devices. We then address the constraints as a discrete problem and, based upon our solutions, we design the features and functionalities of the system. Our modified approach is as follows:

1. Identify goals
2. Collect usage data
3. Formalize pattern from data
4. Address constraints
5. Design features and functionalities
6. Build and test
7. Visualize user experience

In addition to the slight modification of the earlier phases, we added a visualization step as an aide in the iterative process. By visualization, we refer to the diagrammatic mapping of the user experience in order to see the experiential flow. We examine this in the next section.

Visualizing the Mobile User Experience

Designers need to have a global view of the applications they design but, in standard desktop applications, large feature sets often make this an unwieldy task when applied to user experiences. On the other hand, mobile phones are extremely constrained and, in such systems, it is not unrealistic to attain a global view over all the possible user experiences.

The mobile experience diagram—or mobex for short—arose from a desire to be able to visualize the various paths users can take and what consequences their choices will engender. Such a tool allows us to better control the user experience and hence avoid unforeseen states where the user is confused and frustrated.

We combined elements of flowcharts (for visualizing program flow) with elements of finite state automata (for visualizing simple state machines) to create a diagram tool that has states, transitions and control flow. Key elements of a mobile experience diagram, or *mobex*, include:

- State node – a circle with a label inside to indicate what the user sees;
- Transition arrows – an arrow connecting one state node to another, with at least one label, which is what the user does to trigger the transition (i.e., input) and optional commands to indicate what the system executes in response to the transition in addition to enter another state node;
- Programmatic arrows – a box indicating commands that are executed without user intervention and which may have multiple transition arrows connecting it to other state nodes; and
- Initiation arrow – an arrow with a filled arrowhead point to the state node that defines the start of the user experience.

Figure 1 shows a very simple mobex for a flashcard application. The start state is indicated by a filled arrowhead. Any programmatic settings are given as labels on the arrows. Here, after *k* is set to 1, we enter state F_k, referring to the fact that the front of flashcard *k* is shown to the user. When the user presses the *Next* key on the mobile phone's keypad, as indicated by the *N* on the transition arrow leading out of the F_k state node, a new B_k state is entered, indicating that the user is now shown the back of flashcard *k*. If the user presses the *Next* key again, the transition arrow from node state B_k is followed returning the user to state F_k, which now shows the front of the next flashcard since the index *k* has been incremented by 1, as indicated by the label associated with the transition arrow from B_k.

Figure 2 shows a slightly more complex diagram. We will return to how mobex diagrams can be used in practice when we examine how our application led to unexpected user experiences.

Study Tools on Small, Mobile Devices

We now turn to a case study which illustrates the user-interface design methodology described earlier as applied to a study tool for small, mobile devices.

When we launched the Mobile Prep mobile phone study application in a pilot study at a number of California campuses, including Stanford and U.C. Berkeley in March 2005, we believed that law students would comprise a large part of our user base. However, feedback indicated that, while law students do have a lot of material they needed to commit to memory and would like to use mobile phones as tools to that end, the nature of the material did not lend themselves well to our flashcard tool.

Legal procedures, which are typically presented in outline format, does not lend itself well to flashcards not only because of the volume of information but also because there is no inherent need for a “back” of the flashcard as no relationship exists to take advantage of that. However, a strong need for a memorization tool was expressed and so we searched for solutions to address the needs.

Creating the User-Interface for m-outline

For law students, financial analysts and generally students from any field that requires the memorization of structured outline material, we created a tool called m-outline. We examine the user-interface design phases for creating this solution.

Phase 1: Identify Goals

Our goal was to create a tool for small, constrained devices such as mobile phones, that would facilitate in the memorization of content presented in a structured, outline format. We would want the tool to be easy to use and offer the sort of features that would minimally account for the way students typically study paper outlines. Having established our goal, we were unable to envision much more without studying the needs of the users given the nascent state of the technology and the consequential lack of information on user preferences and habits.

Phase 2: Collect Usage Data

We created a focus group of 9 users and made informal observations of their study habits when they are memorizing paper-based outlines. Informal here refers to the fact that we would let them choose how and where they studied—e.g., libraries, cafes. This is in contrast to user studies that would include recording videos of user behavior and requesting that they verbalize their thoughts. Occasionally we would interrupt to ask questions but we generally left questions until the end to ask them to describe the manner in which they felt they were studying the material.

We focused on how they made use of the hierarchical structure of the outlines, whether there were critical aspects of the paper medium that may not lend itself well to a mobile device, and if they thought they had particular approaches of memorizing the material worth noting.

We observed that the material usually consisted of a considerable amount of text. No diagrams were encountered due largely to the nature of our focus group. Also, while outlines do not have associations as flashcards do, the hierarchical structure of the outlines encode relationship information that is critical.

Phase 3: Formalize Patterns From Data

Prior work exists on navigating through dense data by using collapsed structures but our observations for this type of content revealed particular study behavior possibly more appropriate for memorization than for browsing. We wanted to mimic such patterns, which included:

1. Linear read of the content, item by item – the student commits the content to memory by reading the outline much as a book

2. Direct jump to a particular item – the student jumps to a particular item, usually from recognition of the specific item of interest on the pages or through visual cues such as a highlighted section or the item designation (e.g., §1.3.4)
3. Skim and focus – the student skims the items in linear fashion without necessarily committing them to memory or even reading them entirely (either because they are not interested in those items or because they have already memorized them) before stopping at a particular item and focusing on it for memorization
4. Skip through headings – the student views all the items on the same level but skips all the subsections of those items

Knowing the patterns will evolved into features, we then address the constraints.

Phase 4: Address Relevant Constraints

Given the importance of the hierarchical nature of outlines, it is imperative that we preserve the structure, which is typically encoded by indentation. Since the screens on mobile phones are already very small, our challenge was to find a way to capture the hierarchical nature of the content while making optimal use of the screen real estate. We chose not to indent the text—so that we may fit as much as possible on the screen—but to indicate each item's relative level by coloring the background with “indentation bars” that would provide the necessary visual cues.

We also wanted to consolidate the navigation of the content into as few keystrokes as possible given the cumbersome nature of small devices. Our challenge here is to prioritize the features and map them onto button sequences that would allow immediate gratification as well as a gentle learning curve that would introduce users to more sophisticated features. Virtually all handsets come with two programmer-definable “soft” buttons. We chose to map the escape “Back” key to one and a notion of the “Next Item” onto the other. The directional pad, which offers 4 directional and 1 select key) was also employed but the user is given the ability to fully use the system with no prior training and on just the one soft button. The simplicity is meant to establish off immediate gratification and boost their confidence in using the application.

Phase 5: Design Features and Functionalities

We map out the functionalities in accordance with the patterns we formalized from Phase 3. In particular, we provide users with the ability to access items in the outlines in a linear fashion, in a direct fashion, and by the level of the sections.

The users have the following ways of navigating through the outline, utilizing:

1. Run – the user presses [Next] on the keypad to advance through the outline, item by item
2. Focus – the user can highlight any item by pressing [Left]; subsequently, the user can jump through the highlights sequentially by pressing [Fire]

3. Skim and Stop – the user can scroll ([Up] and [Down]) through the outline in *summary* mode and expand any item by pressing the [Next]
4. Overview – each item can be collapsed (to hide all its subsections) or expanded (to expose all its subsections) via [Right]; this option behaves like a typical graphical file system viewer
5. Ignore – each item can be "lowlighted" by pressing [Left]—which toggles through highlight, normal and lowlight states for an item—so that a Run traversal will skip them; unlike flashcards, they are not deleted because the context of where each item is placed relative to the other items is important to preserve in outlines

Further, we offer three views modes, which are toggled through with the [Right] key:

1. Full outline mode – all the items are fully expanded
2. Summary mode – all the items are represented in the display with only their first line
3. Tree mode – items below a particular level are hidden

Phase 6: Build and Test

We created the prototype and offered it to 6 of the 9 members of the focus group as well as 4 new members who did not participant in the usage data collection phase. We are still in the process of compiling the results but we have learned so far have been encouraging. User comments ranged from "great feature" to "I will definitely use it" to feature suggestions such as creating labels for each item for easy reference. However, there were problems: multiple users observed how confusing the Ignore feature was, resulting in an unpleasant user experience.

Visualizing the User Experience

Though the Ignore feature seemed like a useful idea, the user experience was potentially confusing. Specifically, the application was designed to ignore all subsections if a section were to be ignored. The reasoning was that, if a user knows a section well enough to ignore it, then all the subsections can also be ignored.

While the reasoning appeared sound, it turns out that users complained about being completely lost when they used that feature. It turned out that the visual change appeared so massive on a small screen, it made the implicit hiding of subsections disorienting. We observed a number of users trying to negotiate with that feature and it was evident that it was indeed a confusing feature.

Given the constrained visual environment, re-creating alternative options was tedious and of somewhat confusing. We created a mobex to visualize the various states the users can get into when dealing with the Ignore feature. We limited the mobex to the features connected to the

[Left] and [Right] keys (highlight and view mode, respectively) to simplify the diagram. We found that, no matter what approach we took, the mobex, as shown in Figure 3, displayed a complexity that was beyond our expectations for how the user may navigate the system.

Upon closer inspection, we noticed that one particular state had more incoming transition arrows than all the rest of the nodes. This wasn't apparent when the system was designed, given the numerous combinations implicit in the states. However, the mobex clearly indicated that there was a possibility that one particular state was causing a certain level of asymmetry. Since none of the states have any inherent priority over any other states, we decided to try to remove that extra transition.

The resulting mobex, as shown in Figure 4, was far more symmetrical, as would be expected in such a symmetrical system. While symmetry and cleanliness may not guarantee intuitive user experiences, they are indicative of a system that is, to an extent, predictable and orderly, which are factors that impact upon usability of applications.

The change, once visualized, required only minor re-coding of the Ignore feature. The resulting Ignore feature fared far better in user tests. Although it was arguable that the system was less efficient without the implicit collapse of subsections when a section was ignored, the users all tended to prefer the slower but predictable model over the faster but disorienting model. In this, the mobex diagram proved itself valuable as a tool to detect and remedy potential states of confusion for the user.

Figure 5 shows some screen shots of the completed application.

Conclusions

In our interest in creating study tools for students on small, mobile devices we found that traditional user-interface design techniques needed to be adapted to accommodate not just the physical constraints but the newness of the field, which provided little usage data we could leverage to assess functionalities early in the design process. By making slight variations on the standard methodologies we are able to better approach the design of mobile interfaces in a structured manner. We found that the ability to visualize mobile experiences graphically using mobile experience diagrams to be viable for small, constrained systems and valuable in detecting flaws in the design. Based upon the approach, we successfully created a study tool for small, mobile devices for use with structure outline content that advanced students may want to commit to memory.

References

Ballard, B. (2005). *User Interface Design Guidelines J2ME MIDP 2.0.* 2005, Little Springs Design.

Buyukkoten, O., Garcia-Molina, H., and Paepcke, A. (2001). *Accordion Summarization for End-Game Browsing on PDAs and Cellular Phones.* Proceedings of SIGCHI '01. 213 – 220.

Quesenbery, W. (2001). *Applying a UCD Process to Implementing a UCD Process.* Proceedings of the 48th Annual Conference, Society for Technical Communication, 2001.

Shneiderman, B., and Plaisant, C. (2005). *Designing the User Interface, 4th Ed.* Pearson Education, Inc.

Stone, D., Jarrett, C., Woodroffe, M., and Minocha, S. (2005). *User Interface Design and Evaluation.* Morgan Kaufmann Publishers.

Author Note

James Wen, formerly of IBM Research and amazon.com, is a user-interface specialist focused on applying leading edge technologies to consumer products. He received his degrees from Cornell University and Brown University and has held visiting positions at Oxford University and the Royal Melbourne Institute of Technology. He is president of Positive Motion, Inc., a mobile education company.

Table 1

Insert Table Title Here

Figure Captions

Figure 1. Simple mobex diagram

Figure 2. Slightly more complex mobex diagram

Figure 3. Very messy mobex indicating a troubled user experience

Figure 4. Clean mobex

Figure 5. Screen shots of the m-outline study tool

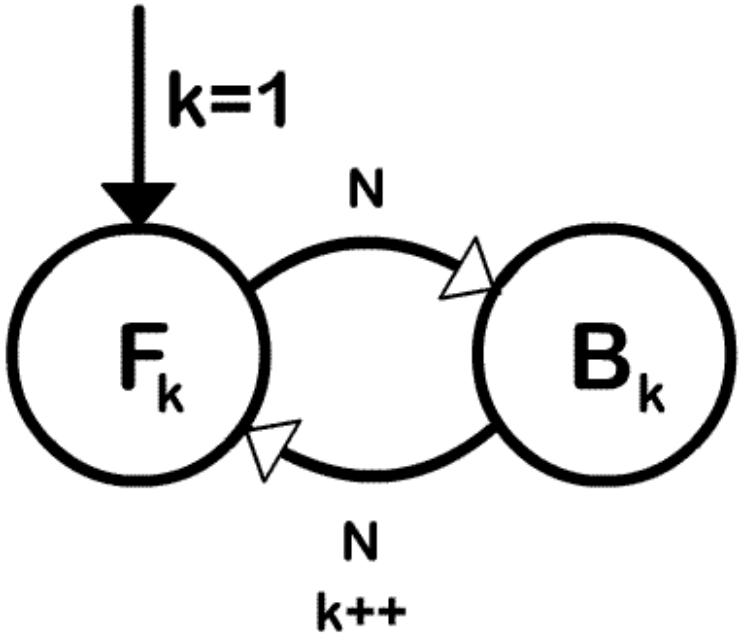

Figure 1

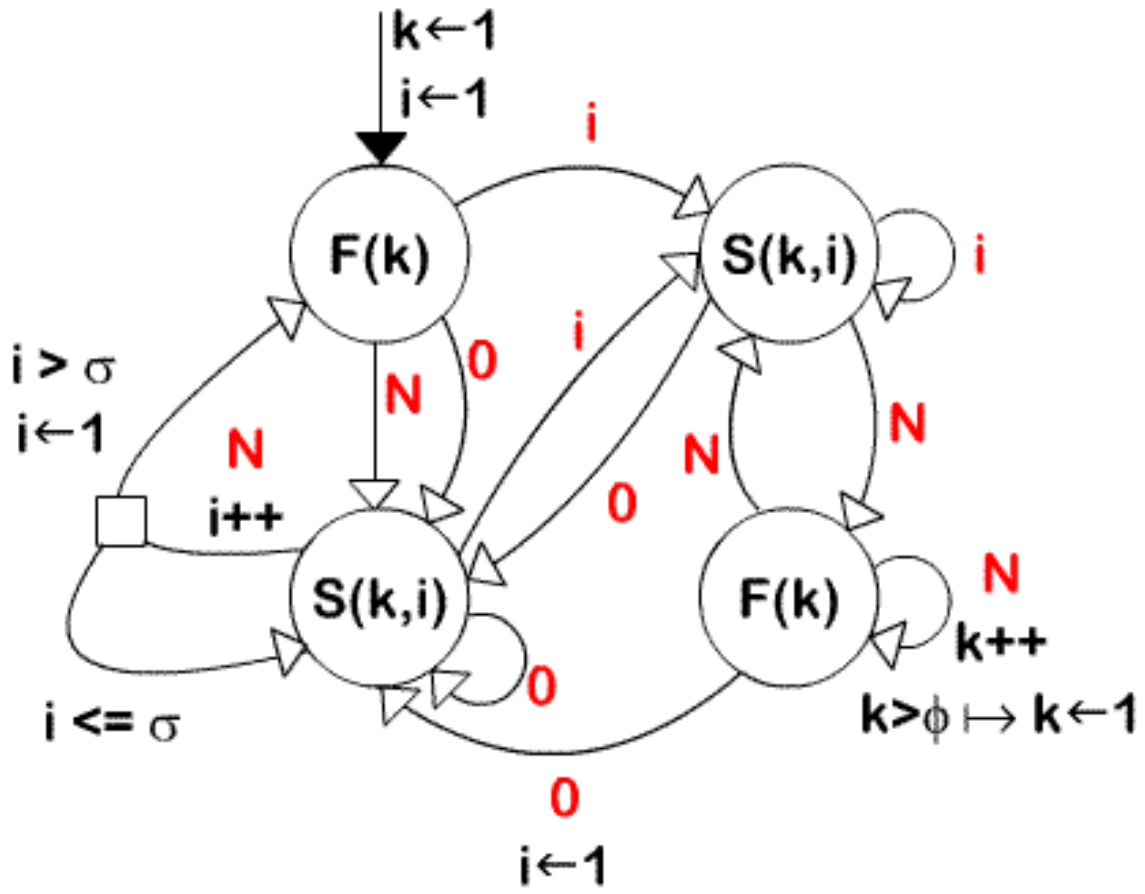

Figure 2

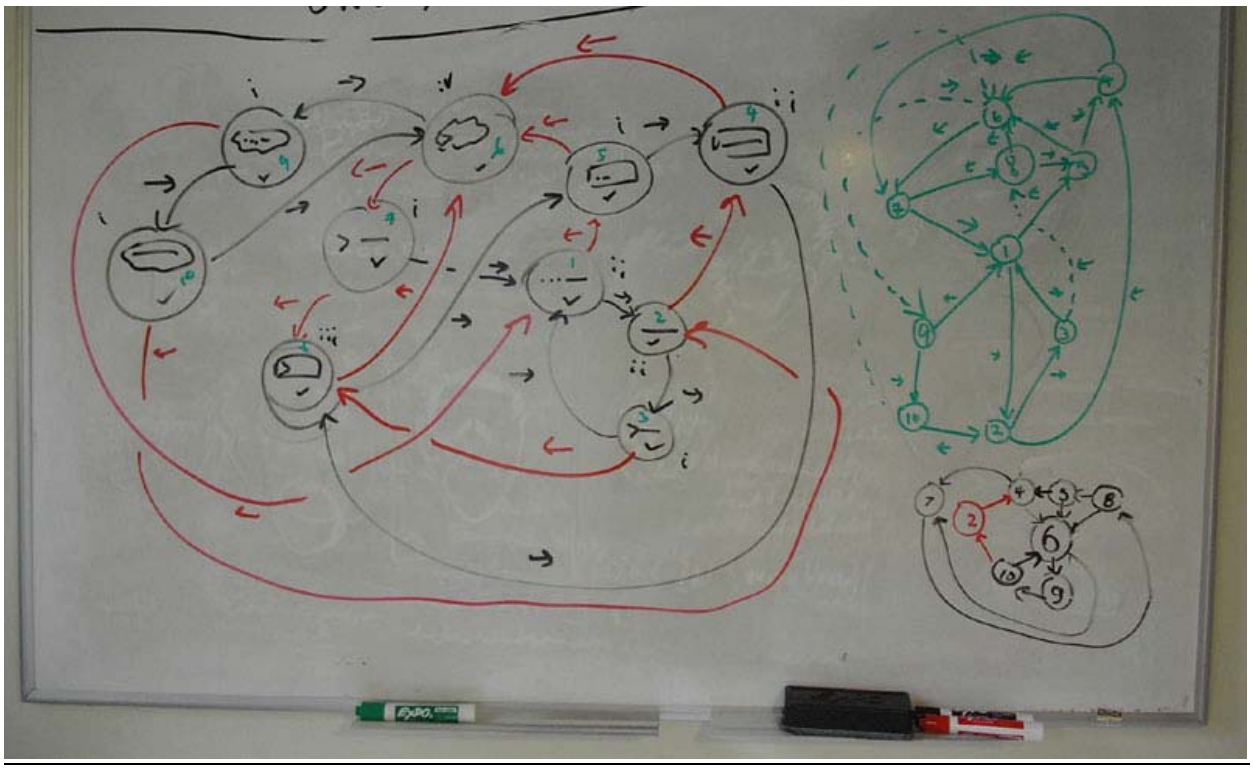

Figure 3

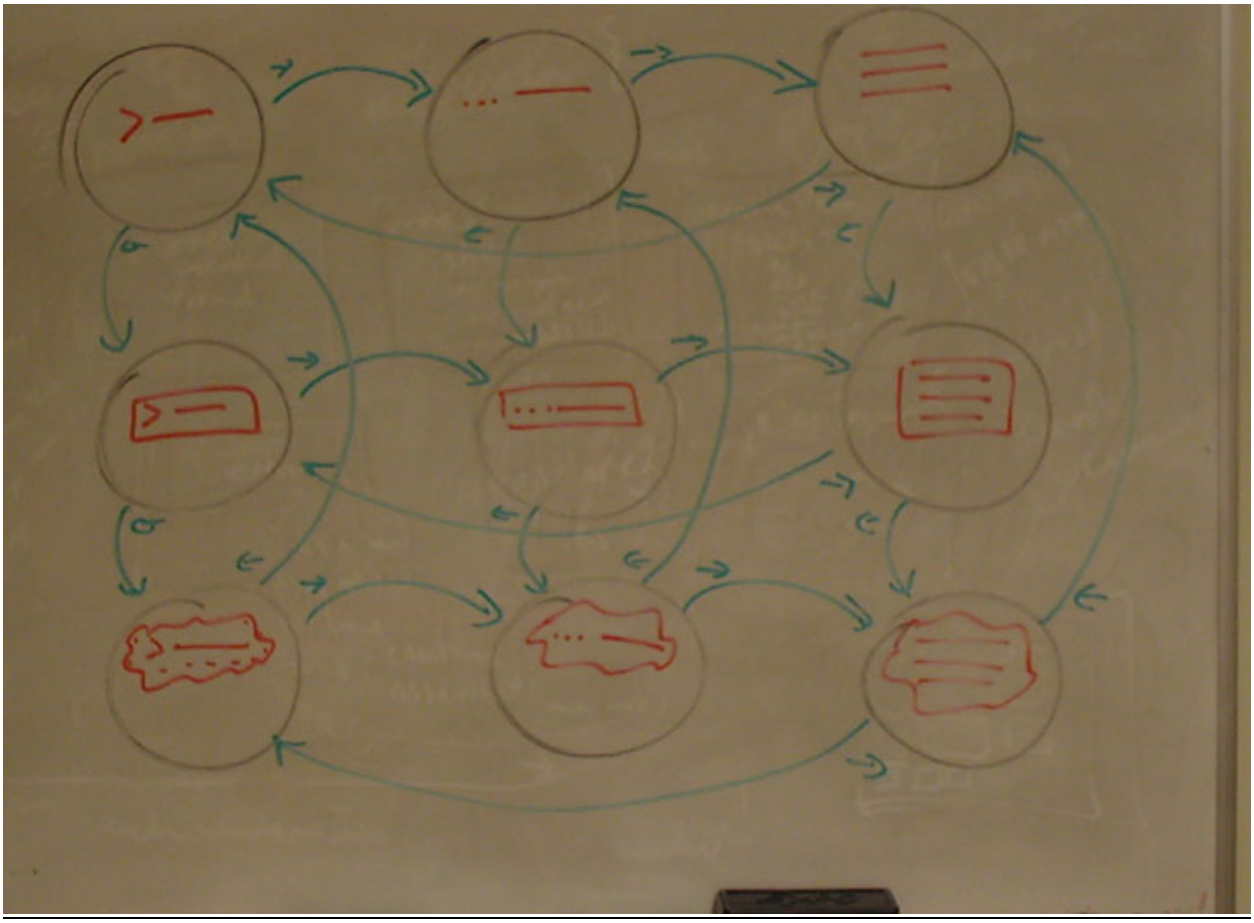

Figure 4

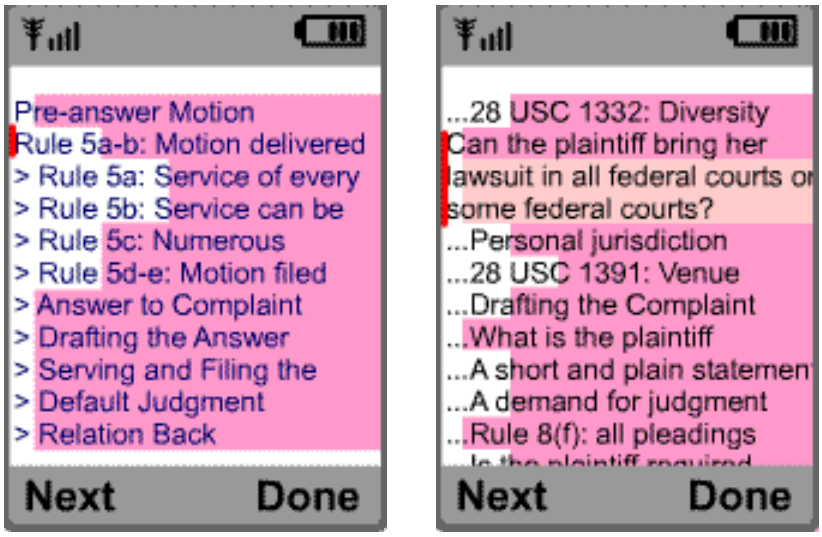
Pre-answer Motion
Rule 5a-b: Motion delivered
> Rule 5a: Service of every
> Rule 5b: Service can be
> Rule 5c: Numerous
> Rule 5d-e: Motion filed
> Answer to Complaint
> Drafting the Answer
> Serving and Filing the
> Default Judgment
> Relation Back
Next
Done
...28 USC 1332: Diversity
Can the plaintiff bring her
lawsuit in all federal courts or
some federal courts?
...Personal jurisdiction
...28 USC 1391: Venue
...Drafting the Complaint
...What is the plaintiff
...A short and plain statemen
...A demand for judgment
...Rule 8(f): all pleadings
Next
Done

Figure 5