Understanding Usability: Investigating an Integrated Design Environment and Management System

Jason Chong Lee[1], Shahtab Wahid[1], D. Scott McCrickard[1], C. M. Chewar[2], Ben Congleton[3]

[1]Center for Human-Computer Interaction and Department of Computer Science, Virginia Tech, Blacksburg, VA 24061-0106
Email: {chonglee, swahid, mccricks}@cs.vt.edu
[2]Department of Electrical Engineering and Computer Science, United States Military Academy, West Point, NY 10996
Email: christa.chewar@usma.edu
[3]School of Information, University of Michigan, Ann Arbor, MI, 48109-1107
Email: bcx@umich.edu

Decades of innovation in designing usable (and unusable) interfaces have resulted in a plethora of guidelines, usability engineering methods, and other design tools. However, novice developers often have difficulty selecting and utilizing theory-based design tools in a coherent design process. This work presents the first extensive usage evaluation of an integrated design environment and knowledge management system for student developers, LINK-UP. The key to this effort is the central design record (CDR), a design representation meant to prevent breakdowns occurring between design and evaluation phases—both within the development team and during design knowledge reuse processes. The CDR is intended to make designs coherent and understandable, thus supporting a principled, guided development process critical for student developers. Three case studies from the classroom illustrating novice designers' use of LINK-UP are presented. A design knowledge IDE incorporating a CDR can help novice developers craft interfaces in a methodical fashion, while applying, verifying, and producing reusable design knowledge.

Keywords: Usability engineering, design tools, knowledge management, central design record, LINK-UP

1. INTRODUCTION

Many different usability engineering methodologies and techniques aim to bring some structure to the design process and allow developers to develop these emerging systems to satisfy the needs of end users. However, it is not always clear how to converge the myriad usability processes and techniques into a coherent, iterative development process, particularly for novice or student developers who may have little to no experience at applying those techniques. This can lead to breakdowns in the usability engineering process where developers are unable to relate user requirements or performance observations back to the design or to arbitrate design goals or intentions with project stakeholders.

Prior work by usability engineers and human-computer interaction (HCI) researchers have uncovered places and situations where breakdowns in the usability engineering process can occur, as well as factors to consider to prevent breakdowns. In addition, research and development in HCI and usability engineering continues without any direct way to leverage this growing body of design knowledge within a development process. This motivates work on *LINK-UP*, an integrated design environment (IDE) and knowledge management system. LINK-UP supports a principled, structured usability engineering process and provides the guidance needed to prevent process breakdowns while enabling developers to access and contribute to an active body of design knowledge as they carry out design actions. While previous papers have reported on the design of LINK-UP (Chewar et al, 2005, Payne et al, 2003), this paper introduces the central design record (CDR), a design representation that makes explicit where and how handoffs—such as the one between evaluation results and the subsequent redesign of a system—occur in the development process and highlights design decisions that need reconsideration during the next development iteration. It is a structured organization of design knowledge (derived from the knowledge library or newly created) that describes

critical features of an interface, what problems the features address, and how evaluation and collaborative design information relates to and affects those features. In this way, it makes explicit the relationship between design goals and the developed interface. For instance, the CDR allows developers to determine links between identified problems and how and an interface design addresses them, or to see relationships between evaluation data and different aspects of the interface. Student designers will be able to see how different design techniques affect designs and why those techniques might be used.

This paper documents three case studies from classroom use that illustrate how novice designers used LINK-UP to engineer interfaces. Building on past experiences with knowledge repositories and principled process-oriented development, LINK-UP serves as a culmination and realization of the prior work of McCrickard and Chewar (McCrickard et al, 2003; Chewar et al, 2005; Chewar and McCrickard, 2005; McCrickard and Chewar, 2006). The results suggest that a design knowledge integrated development environment (IDE) that incorporates the processes and principles of the central design record, can help novice developers apply a design methodology to develop interfaces in a guided, methodical fashion and avoid process breakdowns. Furthermore, there is evidence that, through the use of the CDR, iterative application and verification of reusable design knowledge—important in the education of novice interface developers—is made practical.

2. MOTIVATION AND RELATED WORK

Various methodologies and techniques within HCI have identified problems and issues that need to be addressed in a usability engineering process. This section introduces foundational ideas that motivate the development activities supported by LINK-UP and the central design record.

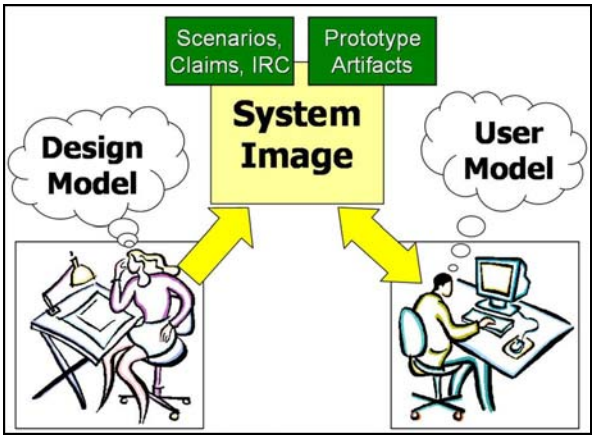

Figure 1. Norman's conceptualization of the system image. The developer captures design goals in a prototype or working system, enabling the study of user interaction with the system.

Norman's theory of action proposes that the design process should acknowledge the existence of three critical components: the design model, user's model, and system image (Norman, 1986). The *design model* is an intended form of the design upheld by the designers. The *user's model* is based on the user's understanding of the *system image*, the physical system and its accompanying documentation (see Figure 1). The design model leads to the development of a system image which is then evaluated by users to produce a user's model. Developers work to converge the two models through iterative development so resulting system designs match user needs and expectations. Critical to this idea is the ability to gauge the convergence such that designers know when they have achieved their goals.

The ability to determine the initial goals and gauge efforts requires an understanding of core concepts behind the system being designed. Wixon stressed the need to focus on engineering-relevant criteria to determine design success both in terms of practice and business (Wixon, 2003). Newman advocates the use of *critical parameters*—measures of performance specific to a class of systems that can determine overall success of a design—as one way to address this need (Newman, 1997). These measures can focus development efforts on the most important parameters of success in an iterative development process. Thus, designers must understand how critical parameters can be used as targets during evaluations.

Usability engineering is concerned with developing interfaces that users can use effectively. It encompasses factors such as learnability, efficiency, memorability, errors and user satisfaction (Nielsen, 1993). Numerous textbooks in usability engineering share many similar characteristics such as user-centeredness and iterative design coupled with analytic and empirical evaluations (Rosson and Carroll, 2002; Hix and Hartson, 1993; Cooper and Reimann, 2003).

One such approach is Rosson and Carroll's Scenario-Based Design (SBD), a design process in which scenarios, narratives describing a particular task, are used in conjunction with design knowledge components called *claims*, which encapsulate the positive and negative effects of specific design features as a basis for creating interactive systems (Carroll and Kellogg, 1989; Rosson and Carroll, 2002). Unlike other knowledge capture mechanisms like patterns and use cases (Borchers, 2000; Carroll and Rosson, 2005), claims provide compact, designer-digestible packets of knowledge ideal for use in class discussions, homework, and activities. They also have the benefit of making trade-offs in design features explicit. Claims rely on scenarios to provide grounding in a realistic situation. The following example scenario, based on the Notification Collage (Greenberg and Rounding, 2001), describes the use of a virtual notice board to allow users to maintain awareness of people they work with:

Pascal, a graduate student, is working on a paper related to his research. While working on the paper, he also wishes to be informed of research related information that is being shared within his lab. He uses the Notification Collage

(NC), which runs on his second monitor, in order to be constantly aware of such information. Pascal can now casually glance at the NC every once in a while in order to see the posted items. When looking at the NC, he visually scans the randomly placed most recent items that are on top. As he looks at the various types of information posted, he gains an understanding of the information contained in the items that are completely visible, but does not know if the information is recent. Knowing that he must find out at a later time when the information items were posted, he returns to his research paper.

The following claim describes a specific design feature of the Notification Collage. Claims such as this can help designers and other stakeholders to consider different design tradeoffs throughout the development process:

Feature: Information artifacts haphazardly posted in an unorganized fashion onto a public display for relevant information delivery, similar to how fliers are posted on a bulletin board.

+ allows users to gain an understanding of an item's age/applicability with respect to the number of items that may be covering it

+ the lack of information categorization accommodates a wide range of different types of information to be conveyed through the display

- BUT overlapping items due to the lack of organization can hinder efforts to read/see a particular information item

- BUT although the relative age of an item that appears on top is newer, the actual age of an item is not apparent

During the process, designers must be able to know when certain design activities, such as the design of a particular task, is completed and how they can test particular aspects of their design. Without such support, a designer may not be able to judge when a development iteration should proceed to an evaluation phase or how the evaluation must be conducted.

The creation of a detailed design representation in SBD is imperative to the design process, allowing explicit analysis of the new system in its anticipated context of use. *The designer's goal is to complete this design representation with as much detail as possible.* With a well-defined design representation, the reuse of design knowledge can prove to be immensely helpful as they are more likely to fit into the structure of the design representation and contribute to the overall design. Being able to effectively store and reuse design knowledge is an active area of study for design domains (Gamma et al, 1995; Majchrzak et al, 2004; Sutcliffe, 2000; Sutcliffe and Carroll, 1999). For example, software engineering community has long advocated reuse of both code and general code architecture solutions through patterns. Within HCI, Sutcliffe and Carroll worked on a framework for documenting and organizing claims in a knowledge repository, although as of yet they have not developed the actual library or tools to support claims reuse processes (Sutcliffe and Carroll, 1999).

Another important consideration in usability engineering is to support communication among the different groups of stakeholders that may be involved in a development process. Given the interdisciplinary nature of usability engineering and HCI, people from different backgrounds may not have an easy way to discuss and reflect on designs. Borchers advocated the use of formally defined patterns to support communication among stakeholders (Borchers, 2000). In a similar vein, Borchers and Erickson have both advocated the use of patterns and pattern languages as a way to support cross-disciplinary discourse (Borchers, 2000; Erickson, 2000). Sutcliffe pointed to structured claims as a way to delivering HCI research knowledge to practitioners, giving them the ability to communicate through claims (Sutcliffe, 2000).

The emergence and acceptance of the various usability engineering processes within the research community, however, does not encourage their acceptance within the industry. Winograd argues that there is a need for design environments to support these and other software system development processes—beyond those provided by programming environments—to support communication and activity flow within the design process (Winograd, 1995). Such systems can allow developers to better use and integrate usability design processes such as those developed by Rosson and Carroll (2002). Design tools have been developed to address this need (Bailey et al, 2001; Lin et al, 2000; Wahid et al, 2004). For example, Bailey et al, have developed a sketch-based interactive multimedia storyboarding tool that also supports behavioral design. Other tools and design processes have been developed to leverage design knowledge, often through patterns (Borchers, 2000; Erickson, 2000; Lin and Landay, 2002). For example, Lin and Landay have developed a system that stores sketched design patterns to support cross-device interface development.

Based on these prominent ideas and implications emerging from HCI research, requirements for LINK-UP were derived. Performing a decomposition of the overall goal of providing useful support for iterative application and verification of reusable design knowledge, the following focus points with respect to the development of design knowledge IDEs are derived:

1. IDEs must support design goal formation and facilitate continuous estimation of design progress through comparison between design goals and resulting design artifacts.

2. To effectively guide specific, incremental design improvements, the system should help developers craft a design representation that is sufficiently detailed to focus evaluation activities.

3. IDEs should facilitate communication efforts among stakeholders around the design representation and its resulting development and evaluation.

4. An interface development support system must be flexible enough to support design and evaluation activities.

3. LINK-UP

To address the concern of providing tool support to designers and achieve the vision described in the previous section, the researchers have developed the LINK-UP system. (A demo of the system is available at http://ticker.cs.vt.edu/LinkupDemo) The system supports the use and reuse of design knowledge and integrates the notion of critical parameters as a guide to designing and evaluating systems. The introduction of LINK-UP continues with a review of the application domain the system focuses on as well as a summary of the key components and features of the LINK-UP system, focused on the Central Design Record module.

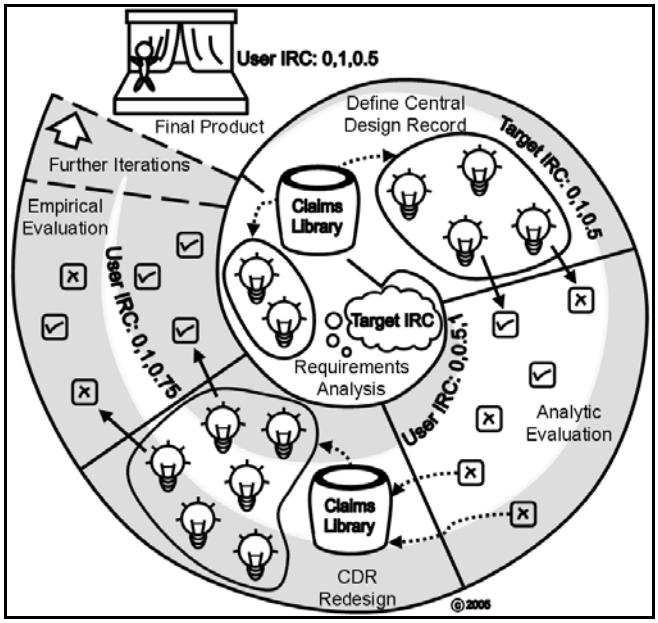

Figure 2. LINK-UP's iterative design process and CDR development. Designers start at the center identifying requirements and a target IRC goal. Design iterations through CDR includes design claims which are tested through evaluations leading to convergence of the design and user models.

Notification Systems

To aid in the definition and application of critical parameters, the LINK-UP system is currently focused on a specific class of interface: notification systems. Successful use of LINK-UP in developing these systems will motivate further research to other domains. In these types of interfaces, a user acts within a primary task while explicitly or implicitly monitoring information through a notification system as a secondary task (McCrickard et al, 2003). Thus, the dual-task nature of these systems is a defining characteristic of the user interaction with these interfaces. In this case, the goal of the notification system is to deliver valued information without introducing unwanted interruptions to the primary task. Instant messengers and e-mail alerts are common examples of such systems. Ambient displays, large screen information exhibits, and car navigation systems are examples of off-the-desktop notification systems.

The domain can be organized by three critical parameters that define innate aspects of notification systems—systems strive to support differing user experiences that can be abstracted in terms of psychological effects (McCrickard et al, 2003). Each critical parameter characterizes the prominence of a psychological effect caused by the design. *Interruption* (I) is the reallocation of attention from the primary task to the notification in the secondary task. A *reaction* (R) is a response to the stimuli to determine whether the notification should be further pursued. Finally, *comprehension* (C) describes the process of understanding the notification and storing the information in long-term memory. Together, these three critical parameters form the IRC framework. The framework uses IRC values ranging from 0 to 1 for every critical parameter. Interface developers can define design goals in terms of these critical parameters and use them as usability and performance metrics for comparison during evaluations. For example, an alarm might have IRC ratings of I:1 R:1 C:0. This indicates that the alarm should be highly interruptive so it is more likely to attract the users' attention, support high reaction so users can respond to the notification quickly and support low comprehension since alarms do not require users to understand and remember details about alarm events.

LINK-UP Development Process

With the four focus points in mind, the goal was to create a system that supports the design of notification systems through the use of critical parameters and Carroll's notion of claims from scenario-based development. LINK-UP consists of a design knowledge repository and modules in which design activities are carried out. The repository, or the *claims library* (Payne et al, 2003), contains claims related to the notification system domain. Designers can search for reusable claims applicable in their own designs by using various searching or browsing features (Sutcliffe and Carroll, 1999). This basis, a structured collection of claims, supports knowledge sharing among designer communities interested in the domain. The *central design record*—developed and maintained using web-based design modules—allow designers to organize and integrate knowledge from the claims library as they develop their systems (Lee et al, 2004; Lee et al, 2005). The central design record is based on Norman's theory of action,

specifically his concept of a system image to like designer and user's understanding of a design (see Figure 1). The CDR makes connections between the designer and user models explicit, highlighting areas for improvement and gaps in the design that can be addressed in subsequent iterations using the claims library as a primary source of knowledge. Figure 2 shows how the CDR and claims library are used to iteratively develop interfaces.

3.2.1 The Claims Library

The basic structure of a claim in the library consists of a feature and a list of upside and downside tradeoffs as shown in the earlier example. This structure is extended with additional information in LINK-UP. Each upside and downside is supported by rationale either summarizing results of an observational study performed by the designer who created the claim or providing references to published research supporting the particular tradeoff. An attached scenario provides context for the claim by describing a task in which the claim can come into use. As a whole, the claim is also assigned an IRC value to depict the effect the claim will have on a notification system, allowing designers to discuss how applicable the claim may be to the overall design goals of their system. Such assignments are critical to integrating the concept of critical parameters into design, providing a base upon which claims can be evaluated. The complete structure of a claim in the library allows designers to create sufficiently detailed design representations through collections of claims.

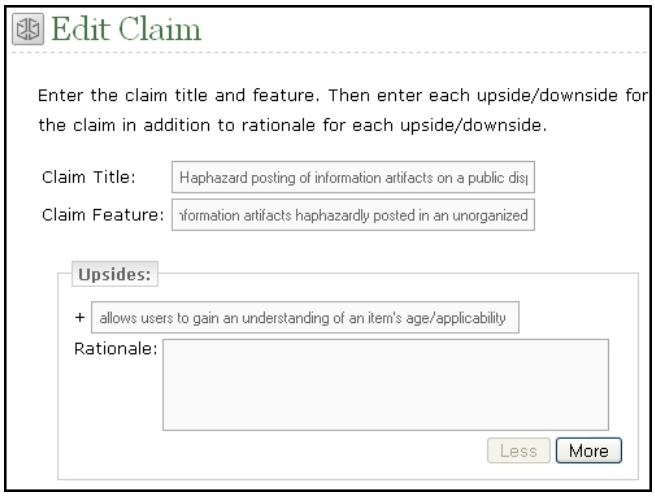

Figure 3. Part of the claim creation process. Designers input a claim title, feature description, upsides/downsides and can optionally input rationale for the claim.

The contents of the library are finite. Hence, designers may not be able to find information that may contribute to their designs. Therefore, contributions to the library are allowed and facilitated through a claims creation and editing process (see Figure 3). Users are guided through a process where they are asked to enter the information for each part of the claim. While this supports current design efforts, future design and reuse activities are also enhanced in the process through such contributions.

The claims library forms the core repository of design knowledge available to LINK-UP users. Design processes to guide developers and support their use and reuse of claims are encapsulated in two different modules: the requirements analysis module and the central design record module (Lee et al, 2005). Guided by these modules, developers construct and manage the central design records for the interfaces they develop. These consist of an organized set of scenarios describing different usage situations, claims related to specific features in the interfaces, and design goals defined in terms of IRC parameters. The CDR acts as a living representation of the design and is used to guide all stages of design including collaborating meetings, evaluations and design improvements.

3.2.2 Requirements Analysis Module

The requirements analysis module serves as an introductory module for designers, and is where they first determine design goals and establish the problems that must be solved. By design, the module is organized into a series of detailed, structured steps (see **Error! Reference source not found.**). This module introduces many of the important concepts and processes within LINK-UP as novice designers make a first pass at defining system requirements before beginning development work in the CDR module. Subsequent changes to requirements or design goals are done in the more open ended CDR module.

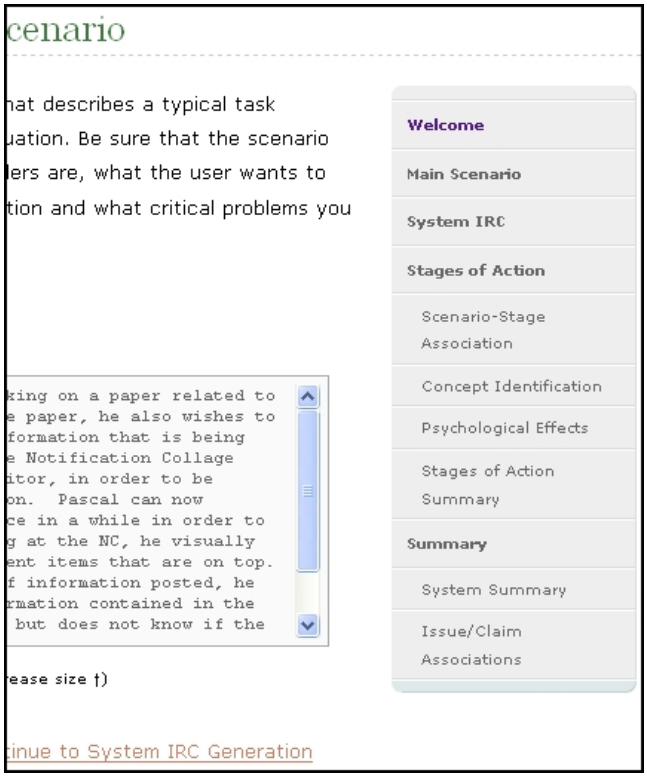

Figure 4. Part of the requirements analysis module. The progress/navigation bar is always visible in the right side of the screen to show designers where they are in the process.

Within this module, the process of scenario-based design is initiated by asking designers to create a problem scenario based on their own analysis. This problem scenario gives insight into the important problems, providing motivation for a new design through a portrayal of current practices. The use of a problem scenario makes it easy for designers to communicate their understanding of the problem domain to stakeholders. These scenarios form part of the CDR and are later linked to design scenarios, which describe how the newly developed interface is used. This relationship—similarly defined between problem and design claims, allow developers to see connections between current practice and how their design affects and improves upon it.

An important part of the design and the CDR is to define design goals in terms of the IRC critical parameters. These goals, called system IRC values, will help developers throughout the design process as they reflect on design decisions and plan subsequent iterations. Furthermore, the processes serve as a formalization of their goals and provide a method for analytic and empirical comparison of intended and actual IRC values. Support for the determination of these system-wide values is provided through a System IRC tool (Chewar et al, 2005) that asks various questions regarding the nature of the system they wish to design.

As a step toward creating the CDR, the problem scenario is decomposed into specific concepts that can later be associated with claims. Each concept is a critical part of the scenario that reflects a problem and need for a solution. The decomposition allows the designer to divide the scenario by Norman's Stages of Action (Norman, 1986) and place concepts within each stage. The stages of action describe human-computer interactions as a cycle between the Gulf of Evaluation and the Gulf of Execution. In the Gulf of Evaluation, users perceive, interpret and make sense of information provided by the system. In the Gulf of Execution, users then form interaction goals, a plan on how to execute that goal, and then physically execute that plan through the interface. Such decompositions help designers understand how users interact with systems and provide a more complete view of user information processing.

Based on the IRC tool, which the developers use to determine the system IRC and questions regarding the desirability of these effects, a stage IRC value is also generated for each Stage of Action. Like the system IRC, these critical parameters help designers focus their design efforts for each stage of action.

As designers progress through this guided decomposition process, they are eventually presented with a list of concepts for which claims are needed. In turn, this prompts the eventual association of design features addressing specific portions of the problem scenario and claims elaborating them with tradeoff expressions. The system facilitates this by offering:

- Access to the claims library to search for reusable claims representing the problems,
- Facilitation of new claims creation to express novel problem or solutions,
- Placement of claims within each Stage of Action.

The stage IRC values allow designers to search for claims that have IRC values close to the required stage IRC values. The requirements analysis module leaves the designer with a specified form of their goals and problems. In the next module, they strive to find solutions to these problems.

3.2.3 Central Design Record Module

Once developers have defined the problem situation and defined design goals in the requirements analysis module, the designer moves on to the *central design record module*. The CDR module is the primary tool through which users develop and manage their designs. Users develop their CDRs by recording design scenarios and claims about the new system, relating them to the defined problem scenarios and claims, and iterating on them as they develop their interfaces. This module is more open-ended and less structured than the requirements analysis module to give users the freedom to change their CDRs as they develop and refine their designs.

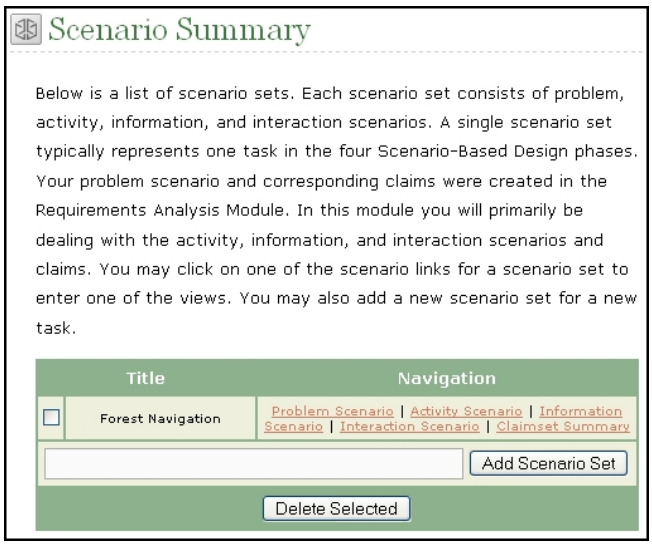

Figure 5. Part of the central design record module. Users create a set of scenarios and corresponding claims for each supported activity. Areas of the CDR module can be accessed in any order.

Designers are first expected to create activity scenarios for each main task they have identified for their notification system. An activity scenario describes the high-level purpose and actions that are to be carried out in a main task (Rosson and Carroll, 2002). Once the scenario is written, designers begin a claims analysis process for the scenario in which they gather claims for the activity scenario and establish relationships to the problem claims identified in the requirements analysis module. Support for the scenario in terms of claims is also broken down by Norman's Stages of Action. A claim is identified for each of the stages in the

Gulf of Execution and Gulf of Evaluation. Just as in the requirements analysis module, the designer can either search for a claim or create a claim. The process of gathering claims within the module supports further claims reuse for designs and encourages the creation of more claims when none are found. A similar process is again followed for information scenarios, scenarios depicting the information a user will encounter during the task, and interaction scenarios, scenarios describing the specific actions the user must take.

The ordering of steps in the CDR module is very fluid. A specific process, as opposed to the requirements analysis module, is not imposed upon the designer (see Figure 5). Many of the steps in the requirements analysis module, such as scenario decomposition, are excluded as it is expected that developers have internalized and understand these aspects of the design process at this point. Designers can switch between working on various parts of their design at any point in time. One can always choose to start creating a new task or to continue developing a previous task. This increases the flexibility of the module and permits the designer to revisit certain parts for redesign as a result of evaluation results.

The various portions of the module support the creation and maintenance of the CDR—a detailed representation of a design that can be used for evaluation purposes. For example, the breakdown of tasks in terms of the stages of action gives designers a chance to evaluate when they are nearing completion of a certain task, but also gives evaluators another perspective on how a certain task is being supported within a design. A breakdown of the design in terms of claims shows the links between prototype features and their corresponding tradeoffs encapsulated by the claims. Imposing such structure on the design in the CDR module forms a gateway to facilitating communication. Stakeholders can discuss certain parts of the design with an established common ground that focuses on specific parts of the module, helping both designers and evaluators reach consensus.

4. CASE STUDIES

In this section three case studies of design projects developed using LINK-UP over several development iterations are presented. Students in an undergraduate Human-Computer Interaction course at Virginia Tech used the IDE in a semester-long project to develop notification systems. Navigation-assisting notification systems with a focus on "off-the-desktop" systems were chosen as a general theme for all the projects. Each project group consisted of 4-5 students. They reported their progress in a series of project documents written at each stage of development. These reports were designed to elicit feedback related to both the system they designed and the process and tools they used to design it. Six reports were written in total by each group, corresponding to requirements analysis, an initial design, an analytic evaluation, a redesign, an empirical evaluation, and a final concluding report. Three exemplar design projects were chosen based on the quality of the feedback groups provided and are reported on below. Specific aspects of LINK-UP and the process it embodies as well as breakdowns and areas for improvement are derived from the observations described in these reports and from the researchers' personal observations throughout the duration of the projects.

Case 1: Huckleberry Trail Attraction Notification System

The first case study documents the development of a notification system running on a PDA that facilitates general enjoyment of attractions on and around the Huckleberry Trail, an outdoor hiking trail. It focused on allowing hikers to discover and learn about various attractions as they walked along a trail without interrupting their general enjoyment of their surroundings. This case illustrates how the CDR can support principled incremental interface improvement by linking evaluation results directly to design decisions.

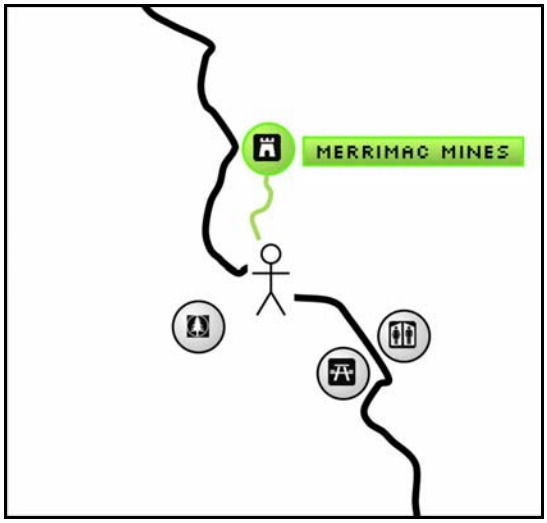

Figure 6. Screenshot of the Huckleberry Trail guide prototype. One area of contention was how interruptive highlighted icons would be to users on the trail.

The group conducted an analysis of the trail and its uses and determined that the trail, built along an old rail line near Virginia Tech, is visited by many people to enjoy its natural beauty. However, many attractions, especially those that are not directly on the trail, are easily missed. Since Virginia Tech hosts many students from different areas of the world, the developers decided to develop a PDA-based tourist guide system that could supply useful information to newcomers about the trail without disrupting their enjoyment of the natural surroundings.

Based on this initial analysis, the designers developed a problem scenario illustrating the need for their design. Using the requirements analysis module, the developers decomposed their scenario to aid in extracting problem claims. They also used the embedded target System IRC estimation tool to estimate the targeted design model IRC

value for their system. Overall, they found that the system should have low *interruption*, because they do not what their system to distract from enjoying the natural environment and should have high *comprehension* so that users are aware of the different sights around them as they travel along the trail. The developers found the overall process of using LINK-UP to be constraining and tedious, though they were able to extract problem claims that highlighted the issues they hoped to address. In addition, the developers found the IDE's problem claim recommendation capabilities to be of little value. This may be because the recommended claims, which are selected by LINK-UP based on IRC values and pre-selected design features, do not encode enough semantic information to return useful claims to the user. An initial design was developed based on the problem scenario and claims. Many design claims evolved directly from relationships to problem claims. This connection between the problem and design space proved useful in later justifying design decisions when evaluating their designs.

In the analytic evaluation, one other group in the class acted as expert evaluators and attempted to identify problems with the design for the Huckleberry Trail notification system. The evaluators were given access to both a paper prototype, which gave a broader user perspective of the design, and the underlying scenarios, claims, and system IRC value organized by the CDR module, giving evaluators access to the design model of the system being developed. The developers noted that the CDR is not an exhaustive representation of the design. Comments from the evaluators uncovered problems that were not considered and recorded in the CDR. For example, the evaluators noted that there was no mechanism to support destinations or sights that were far off the trail.

A redesign of the system was focused on mitigating problems that were identified in the analytic evaluation. A functioning prototype was then developed based on the refined CDR. The prototype ran on a laptop, rather than a PDA, and was displayed next to another laptop that displayed a slideshow of nature-related images to simulate traveling through the trail.

In the subsequent empirical evaluation, users were told to pay as close attention to the images as possible as the notification system ran in the periphery on the second laptop. Specific claims in the CDR further guided the empirical evaluation. The empirical evaluation was meant to verify untested claims in the CDR and determine whether the user model IRC value matched the design model IRC value. In addition, the CDR made the link between evaluation data and the design rationale explicit. For example, in the analytic evaluation, the evaluators thought that highlighting icons on the display may cause too much interruption for the user. The developers decided to try to mitigate this issue by using non-highlighted icons on the system. Both highlighted and non-highlighted icons were tested in the empirical evaluation, and the developers did find that the highlighted icons caused too much interruption (see Figure 6). In addition, the designers compared the use of symbolic and realistic icons in their interface. They found that the meaning of the simpler symbolic icons were more obvious to users since they looked like familiar map icons and were not visually complex. This meant that the simpler icons made the system less cognitively demanding and allowed users to focus on the handheld less. In both cases, the simpler design choices allowed the developers to support their overall design goals which were embodied in their System IRC value. These examples illustrate the link between design rationale in the CDR to empirical results and to the system goal made possible by the persistent representation of the design model that is stored and maintained through the CDR module.

This case highlights the general success of using critical parameters to measure the success of the system at achieving initial goals while using the CDR to direct iterative refinements to achieve them. Similar successes would become apparent through further iterations, more refined prototypes, and field testing on the Huckleberry Trail.

Case 2: Online Dispatcher Notification System

The second case study describes the development of an in-vehicle navigation device for police officers. This system allows officers to determine their current location, the location of fellow officers, and the best route to the alert site. This is meant to mitigate the inefficiency of current radio-based information relay. The study highlights the principled, incremental design improvements made possible through the CDR and the benefits in initially using a guided process to steer requirements analysis.

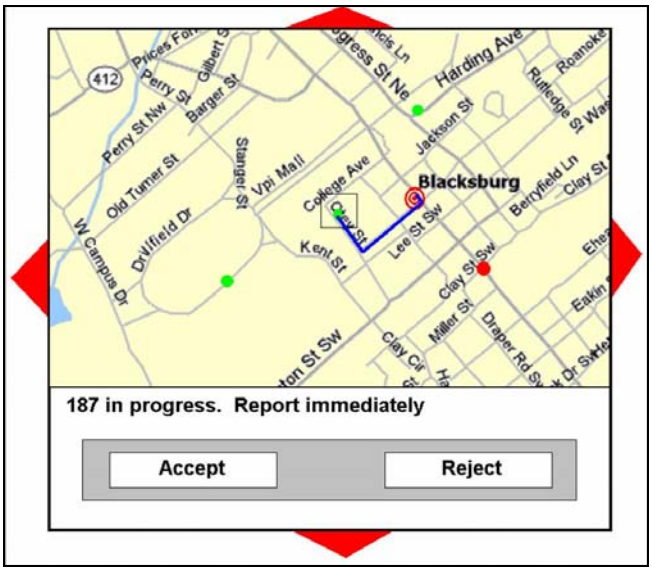

Figure 7. Screenshot of online dispatcher prototype. Redesign efforts will need to focus on making the model more comprehendible during navigation tasks.

These developers reviewed information related to police procedures available on the internet and conducted an

interview with an officer at the local police department. Based on the information, they laid out the high-level goals and tasks that the system should support. By connecting to the central police control center, the Online Dispatcher will notify officers about incidents he or she needs to respond to. It will automatically determine the fastest route to the incident and also indicate where fellow officers are located. They determined that the system should have high interruption to alert officers to people in need of assistance, high reaction so that officers know how to respond to different alerts quickly, and moderate comprehension so officers can maintain awareness of surrounding information and routes leading to affected areas.

The developers found the problem scenario decomposition process in the requirements analysis module to be helpful in separating different features and concerns from which claims could be derived. However, the developers did not have a clear understanding of the stage IRC values and did not use them to find relevant library claims. Similarly, the System IRC estimation tool was helpful in allowing the developers to consider overall goals of their system. Although the group found the overall process to be time-consuming it ultimately aided in their design because it allowed them to develop both a top-down and bottom-up view of their design space.

To minimize disruption to accepted police protocols and encourage acceptance, the designers considered each aspect of the defined problem situation to determine which specific areas to target for design and which to keep the same. For example, although the system is built as a small display and will support visual notifications, it will continue to support audio notifications because officers are already accustomed to such alerts and know how to react to them. This will then allow officers to draw attention to the display for further information or interactions.

The analytic evaluation allowed the developers to identify potential usability concerns. The CDR prompted and focused discussion among the developers and evaluations about important aspects of the design, specifically those related to how interruptive the system is to the officer's primary task of driving. In reviewing the CDR, the evaluators believed that the system may be too distracting for the officer to interact with while driving. A suggested way to mitigate this problem is to require the car to be at a standstill before an officer can interact with the system (e.g. select an alternate route). The developers decided not to limit the system functionality and ultimately did not mitigate this problem because they believed the reduced utility tradeoff for the additional safety feature would be too great. The need for flexibility and control combined with the training and discipline of their system's target users outweighed the need for any kind of safety lock. This demonstrates how the CDR can serve to encourage discussion and critical thought revolving around design features among different stakeholders.

In the empirical evaluation, the dispatcher notification system prototype ran on a laptop in the approximate location it would be in a squad car. In the interests of safety, the driving task was simulated by having the study participants operate a car in a driving simulation game. The designers were unable to recruit actual police officers due to time constraints. Instead, they recruited undergraduate students with several years of driving experience and average experience using online map services. The designed tests were derived from specific design claims in the CDR and were primarily focused on how effectively the system could notify drivers without distracting from their primary driving task and whether drivers were able to comprehend the information provided. For example, they tested things such as how distracting the audio notifications were and how easy it was to respond to a notification while they were focused on driving. One of their design goals was to have a relatively high level of interruption since police officers need to respond to dispatch calls in a timely manner. However, the results of their study indicated that their audio notifications seemed to be too interruptive in that they caused participants to crash their vehicle too often. They are focusing on refining when and how audio and visual notifications are used to provide information to officers. In addition, the participants found it too difficult to follow and understand the map (Figure 7). Despite these setbacks, the developers determined that they need to focus future redesign efforts on the information design claims in their map to support better comprehension and their interruption claims in their CDR. They want to avoid requiring more manual interactions with the system in making these changes so that officers can keep their hands on the steering wheel. The group was confident that such efforts would lead to a design that better matches their initial goals.

This case study demonstrates how the CDR can support focused, incremental design improvements through explicit claims analysis and analysis of evaluation data. It also demonstrates how LINK-UP can help guide developers in specifying features and problems to focus on, particularly in the early stages of design.

Case 3: Motorcycle Navigation Notification System

The third case study documents the design of a vehicle navigation system for motorcycles. This system was intended to provide route information, points of interest, and real-time traffic and weather information. The challenge of this system was in providing these features within the unique constraints of an operating motorcycle (see Figure 8). The targeted users of this system were recreational motorcyclists, who often take back roads and non-optimal paths to maximize enjoyment of the ride. The developers determined that this system needs a moderate level of interruption and a low level of reaction to minimize the risk of distracting the motorcyclist while still providing useful information, and a moderate to high level of comprehension of notified information so that they are

always aware of where they are and of their surroundings in general. The system relies primarily on audio notifications which are supplied to the user through an earphone the user puts on under his helmet. A large display acts as a secondary reference both for notifications and additional information about the route.

Figure 8. Rendering of motorcycle navigation notification system prototype.

These developers differed from the previous groups in that after developing their scenarios, they had success in finding claims in the claims library that relate to their designs. In the problem scenario, the motorcyclist uses a PDA with a GPS navigation system to trace a route before getting on his motorcycle. The scenario decomposition process combined with the stage IRC values in the requirements analysis module helped in finding claims in the library. For example, one reused problem claim describes the benefits of interacting with a hand-held device. Although originally used to describe a remote control for an MP3 player, the developers found the claims were general to be reused to describe upsides and downsides of their problem scenario. Similar reuse occurred in the design phase of their project. In most cases, claims were found in the library that matched existing ideas for their interface, rather than as a way for them to explore how to develop their system. This allowed the developers to access positive and negative effects that they may not have considered had they themselves created the claims. This reuse-first design philosophy allowed the developers to better consider their design options and tradeoffs without sacrificing the creative aspects of interaction design.

The developers did not gain from the analytic evaluation as the designers in the other case studies did. They noted that their evaluators did not have an in-depth understanding of their system and problems they identified were actually addressed elsewhere in the CDR. For example, the evaluators thought the system should include a backlight to maintain visibility of the system at night, but the designers pointed out that this issue was already addressed by another claim in the CDR. Though unfortunate, this highlights how the CDR, with its related scenarios and claims, can be used to act as a common language through which project stakeholders can both discuss the design and resolve any misunderstandings (Borchers, 2000).

The developers also had problems managing the CDR through LINK-UP in the redesign phase of the project. They found that the number of scenarios and claims that had to be managed and updated was daunting. In addition, since a functioning prototype was not assigned to be developed until just before the empirical evaluation phase of the project, the developers were frustrated by working almost exclusively with the CDR. They noted that the CDR records the design model of the interface, but it does not convey how a user actually interacts with it.

Like the previous case study group, the developers ran their prototype system on a laptop in front of the participant as he or she attempted to navigate around on a motorcycle in a driving simulation game. All participants were motorcycle owners and so were members of the designers' target user group. Audio notifications were fed from the system to the participant through an earpiece. These developers did not gain as much insight into potential design improvements as the previous groups in that most of their participants performed within expected parameters for the tasks they laid out. They found that audio-based notifications were an effective way to notify users without distracting too much from their primary driving task. The large display screen and simple graphics were also easy to read and effectively supplemented the audio notifications when the user needed additional information. In addition, they found that their system seemed to support moderate levels of interruption and relatively high comprehension, which were in agreement with their initial system IRC goals. Overall, they found the evaluation on the functioning prototype to have provided the most valuable feedback. In particular, the open-ended feedback suggested that the system would be better marketed toward a specific type of motorcyclist and suggested additional features and aesthetic improvements. For example, participants suggested that the overall size of the system be reduced to be less bulky—perhaps by reducing the size of the frame. Some users also mentioned that the display had some unneeded information and could be simplified further to make it easier for users to glance at the system and get all the information he or she needs.

This case study highlights how reused claims can help in designing interfaces and how the CDR can support better communication among stakeholders. It also suggests how an empirical evaluation module needs to support both specific design issues related to principled, incremental design improvements and to broader issues that may be outside the scope of the IRC framework.

5. DISCUSSION

In this section, results from the case studies are synthesized to draw out conclusions about LINK-UP. Both the

strengths of this approach and areas for improvement are discussed, summarized in Table 1, with respect to the four focus points. These highlight the value of this approach, particularly in educating novice designers, and suggested areas to focus future efforts.

The critical parameters of the notification systems, embodied in the system IRC values, proved to be a valuable guide in supporting the first focus point: IDEs need to support design goal formation and comparison between design goals and the resulting design artifacts. Developers were able to verify specific hypotheses made in newly created claims and speculate on how they affect the system IRC value. The case studies demonstrated the importance of integrating critical parameters into LINK-UP and guide reflection and iteration on the design through the CDR. They proved to both guide design activities and estimate design progress based on evaluation results.

The case studies also suggest the second focus point—IDE aids in design representation formation to focus evaluation activities and specific, incremental design improvements—is also supported. By their nature, scenarios do not exhaustively detail every aspect of an interface in use. Subsequent claims extracted from those scenarios are similarly limited. Thus the CDR is not an exhaustive representation of an interface design. However, this worked to the advantage of the developers by focusing evaluation and redesign activities on specific aspects of the interface—especially the notification task defined in terms of the target IRC values. This supported iterative, risk-driven development by having designers focus on key, high risk aspects of the system first. The direct relationship between the problem and design space, captured in the respective scenarios and claims in the CDR, also encouraged careful consideration of current practices while developing the new system. Planning empirical evaluations partly around individual claims also supported the second focus point. Failures in the defined tests could be linked

Key focus points	Case Observations by Design Phase			
	Requirements & Initial Design	**Analytic evaluation**	**Redesign**	**Empirical evaluation**
IDE supports goal formation and facilitates design progress estimation	(+) Generated IRC is close to expected values *[case 2, case 3]*	(+) Critical (IRC) parameters focus evaluation on non-trivial aspects of design *[case 1, case 2]*	(+) IRC parameters focus redesign efforts on critical design features *[case 1, case 2]*	(+) Evaluations focused on specific features verify hypotheses made in claims *[all cases]*
IDE guides design representation development to support evaluation activities	(+) CDR focuses on specific design features *[all cases]* (+) juxtaposing problem & design space allows careful reflection *[case 1, case 2]*	(+) Redesign efforts focused on features defined to be most important by developers *[case 2, case 3]*	(+) Claims analysis based on evaluation results guide redesign *[case 1, case 3]* (+) Untested or contested solutions can be deferred until empirical evaluation *[case 1, case 2]*	(+) link between design model and empirical results guides redesign *[case 1, case 2]* (−) general feedback not directly supported *[all cases]*
IDE facilitates communication among stakeholders around design concerns	(+) Paper prototype helped make design less abstract by connecting design rationale to interface. *[case 1, case 3]*	(+) CDR provides tangible design model → to support discussion of interface between developers and evaluators *[all cases]*		(+) Conflicts uncovered in analytic evaluation resolved *[case 1]*
IDE is flexible enough to support iterative design/evaluation activities	(+) Scenario breakdown helps to find claims *[case 3]* (−) Requirements process constraining & tedious *[case 1, case 2]* (+) Claims search → supported by critical parameters aids reuse *[case3]*	(−) Hard to manage, comprehend large # of claims and scenarios in CDR *[case 1, case 2]*	(+) Claims format aided in analyzing design tradeoffs *[case 1, case 3]*	(−) functioning prototype (outside LINK-UP) provided most valued feedback *[case 2, case 3]* (+) validated claims can be updated with empirical data *[all cases]*

Table 1. Summary of positive and negative case observations and relation to focus points.

directly to design decisions—expressed in claims. Developers then knew where to focus prototype redesign efforts. The case studies demonstrate the value in supporting incremental improvements through the tight coupling of design representations with evaluation data. Developers were able to understand the nature of iterative development and saw the potential of tightly focused redesign efforts that are more likely to resolve identified problems and complement a user's current work environment.

LINK-UP, and its implementation of the CDR, also supports the third focus point—an IDE needs to support communication among stakeholders around the design representation's development and evaluation. This work shows that by acting as a communication point between developers and evaluators, the CDR encourages discussion and evaluation of the interface design. The paper prototype combined with the goal IRC value gives a high-level view of the system while the CDR encapsulates the design model of the interface. This allowed analytic evaluators to view specific aspects of the system from both a designer and user perspective. In addition, the analytic evaluation helped to resolve misunderstandings as documented in the Motorcycle Navigation System group. Thus, a focused design representation that is encoded in an easily understandable manner allows different groups to better reflect on design decisions and can aid in conflict resolution with respect to the interface design.

As the developers used the LINK-UP system itself, strengths and limitations of various parts of the IDE became apparent. Thus, the fourth focus point—IDE is flexible enough to support iterative design and evaluation activities—is partially supported. The relatively linear, guided process in LINK-UP, particularly in the requirements analysis module, was helpful to novice developers in identifying problem scenarios, claims and in defining design goals. However, all groups found this process to be tedious and restrictive to varying degrees. Only the Motorcycle Navigation group was able to leverage the given search features, including the IRC parameter-based search, to find relevant claims. The other groups were unable to find claims from the library to reuse. The Huckleberry Trail group noted that they did not understand the parameter-based claims search; this emphasizes the high learning curve needed to use a faceted search (Chewar and McCrickard, 2005). However, the claim recommendation system, which was meant to mitigate this problem and was based on the IRC parameters, did not help that group find claims either. The fact that one group was able to effectively reuse claims from the library and a general willingness from the other groups to reuse point to limited search capabilities as the limiting factor. This highlights the need for a multi-dimensional search system that is tightly integrated with an IDE. This should include a basic search field such as those used in popular web search engines. Furthermore, several groups noted the difficulty they had in managing and reviewing the large number of scenarios and claims through LINK-UP. The size of the CDR may cause other issues in the design process such as in analytic evaluations where expert evaluators may have problems reviewing and making sense of all of the CDR information. This problem could be mitigated by limiting or allowing users to prioritize information in the CDR so that only critical features that are the focus of the current iteration are visible.

Overall, the LINK-UP system did help educate students in the scientifically appropriate use of usability engineering processes and knowledge creation methods, albeit while sacrificing time-efficiency and flexibility in the design process. These are not critical drawbacks, since LINK-UP is intended for novice developers for whom the learning of processes and methods is of the highest importance. Most groups found the empirical evaluation with the functioning prototypes to provide the most useful feedback because end users were physically interacting with a real system. In addition, all groups were able to successfully use the CDR while developing their systems to focus design activities, carefully consider tradeoffs, communicate design concerns to stakeholders, and contribute design knowledge into the claims library.

6. CONCLUSIONS AND FUTURE WORK

Numerous usability methodologies and techniques exist to support the development of computer systems and interfaces. However, it is not always clear to novice developers how to use these different processes and techniques in a coherent design process. Through the case studies this research demonstrates that a design knowledge IDE centered on the central design record can help novice developers make connections between requirements data, design representations and evaluation data and better understand how to leverage that information to incrementally improve designs in an iterative usability engineering process. It is also shown that the use of CDR supports the application and verification of reusable design knowledge for novice developers.

Based on these results, the following guidelines are derived for design knowledge integrated development environments for use in educational settings that incorporate a module patterned on the CDR:

- Persistent design representations should support multiple or 'current' perspectives to direct development efforts on salient design concerns and streamline development processes.
- Design rationale should be tightly coupled to evaluation data to show novice developers where to direct redesign efforts and support validation of design knowledge for future reuse.
- Design representations need to be easily understandable, with goal states stated in unambiguous terms, perhaps through critical parameters, to support

collaborations among novice developers and other project stakeholders.

- Design knowledge IDEs should support, but not require, guided processes to aid in knowledge capture, knowledge use and goal formation so that they support developers with varying levels of knowledge and expertise.

LINK-UP is actively being developed to support the education of novice developers (Fabian et al, 2006; Wahid and McCrickard, 2006). Future efforts are focusing on addressing the problems that were encountered during this evaluation. First, the development process will be changed so the CDR will only show information related to important design issues imperative in the current iteration. This will make the CDR easier to manage and to use in collaborative discussions. Second, more emphasis will be placed on developing functional prototypes earlier in the design process so designers can get more accurate feedback from end user evaluations. This will require the tight coupling of the prototypes to the CDR to retain the benefits described in this work. Third, work will continue to find better and more intuitive ways to search for and organize design knowledge from the claims library.

The current implementation demonstrates the potential for a reuse-enabled integrated design environment in guiding novice developers through the interface design and evaluation process. Lee and McCrickard are researching ways to use this research to find ways for software development practitioners to integrate usability engineering practices into agile software engineering methodologies (Lee, 2006; Lee and McCrickard, 2007). These programmer-centric design methodologies have become increasingly popular in the last decade because of their success in allowing developers to deliver working software on time and on-budget (Beck, 1999). However, these methodologies do not account for usability—resulting in software systems that meet development goals but which are often difficult to use. This work leverages the lessons learned in this research, including the use of design representations such as the CDR, and how tools such as LINK-UP should be designed, to develop practices and tools to support agile, multidisciplinary teams in developing usable systems in an efficient manner. These case studies, combined with the resulting guidelines and ongoing work on agile usability will serve as an important first step towards a dynamic, collaborative HCI development environment and knowledge repository that is used and extended by designers from different disciplines both in academia and in industry.

7. ACKNOWLEDGMENTS

We wish to thank Edwin Bachetti, Andrew Jackson, Justin Belcher, Mat Davis, Kai Tang, Tripp Lilley and Jamie Smith for their assistance in the design and implementation of LINK-UP, and the students in the Spring 2005 CS 3724 Human-Computer Interaction course for their extended use of LINK-UP in their activities and course project.

8. REFERENCES

1. Bailey, B. P., Konstan, J. A., and Carlis, J. V. (2001). DEMAIS: designing multimedia applications with interactive storyboards. *Proc. 9th ACM Intl. conference on Multimedia*, 241-250.
2. Beck, K. (1999). Extreme Programming Explained: embrace change. Addison-Wesley Longman Publishing Co., Inc., Boston, MA.
3. Borchers, J. O. (2000). A Pattern Approach to Interaction Design. *Proc. DIS 2000*, 369-378.
4. Carroll, J. M. and Kellogg, W. A. (1989). Artifact as theory-nexus: Hermeneutics meets theory-based design. *Proc. CHI 1989*, 7-14.
5. Carroll, J. M. and Rosson, M. B. (2005). A case library for teaching usability engineering: Design rationale, development, and classroom experience. In *Journal on Educational Resources in Computing.* Vol 5, Issue 1, 1-22.
6. Chewar, C. M., Bachetti, E., McCrickard D. S., and Booker, J. (2005). Automating a Design Reuse Facility with Critical Parameters: Lessons Learned in Developing the LINK-UP System. In *Jacob, R., Limbourg, Q., and Vanderdonckt, J. (Eds.), Computer-Aided Design of User Interfaces IV*, Norwell, MA: Kluwer Academic Publishers, 235-246.
7. Chewar, C. M., and McCrickard, D. S. (2005). Links for a Human-Centered Science of Design: Integrated Design Knowledge Environments for a Software Development Process. *Proc. HICSS 2005*, 10 pgs (CD-ROM).
8. Cooper, A., and Reimann, R. (2003). About Face 2.0: The Essentials of Interaction Design. Wiley Publishing Inc., Indianapolis, IN.
9. Erickson, T. (2000). Lingua Francas for Design: Sacred Places and Pattern Languages. *Proc. DIS 2000*, 357-368.
10. Fabian, A., Wahid, S., Bhatia, S., and McCrickard, D. S. Creating an Interactive Learning Environment with Reusable HCI Knowledge. *Proc. ED-MEDIA 2006*, 2314-2322.
11. Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). Design Patterns: elements of reusable object-oriented software. Boston: Addison-Wesley Longman.
12. Greenberg, S. and Rounding, M. (2001) The notification collage: posting information to public and personal displays. *Proc. CHI 2001*, 514-521.
13. Hix, D., and Hartson, H. R. (1993). Developing User Interfaces: Ensuring Usability through Product and Process. John Wiley & Sons, Inc., New York, NY.

14. Lee, J. C. (2006). Embracing Agile Development of Usable Software Systems. Doctoral consortium paper in *Proc. CHI 2006*, 1767-1770.

15. Lee, J. C., Chewar, C. M., and McCrickard, D. S. (2005). Image is Everything: Advancing HCI Knowledge and Interface Design Using the System Image. *Proc. ACMSE 2005*, Vol. 2, 376-381.

16. Lee, J. C., Lin, S., Chewar, C. M., McCrickard, D. S., Fabian, A., and Jackson, A. (2004). From Chaos to Cooperation: Teaching Analytic Evaluation with LINK-UP. *Proc. E-Learn 2004*, 2755-2762.

17. Lee, J. C., McCrickard, D. S. (2007). Towards Extreme(ly) Usable Software: Exploring Tensions Between Usability and Agile Software Development. *Proc. Agile 2007*, to appear.

18. Lin, J., and Landay, J. A. (2002). Damask: A Tool for Early-Stage Design and Prototyping of Multi-Device User Interfaces. *Proc. 8th Intl Conference on Distributed Multimedia Systems*, 573-580.

19. Lin, J., Newman, M. W., Hong, J. I., and Landay, J. A. (2000) DENIM: Finding a Tighter Fit Between Tools and Practice for Website Design. *Proc. of CHI 2000*, 510-517.

20. Majchrzak, L., Cooper, L. P., and Neece, O. E. (2004). Knowledge Reuse for Innovation. *Management Science 50*(2): 174-188.

21. McCrickard, D. S., and Chewar, C. M. (2006). Designing Attention-Centric Notification Systems: Five HCI Challenges. In *Forsythe, J. C., Bernard, M. L., and Goldsmith, T. E., (Eds.), Cognitive Systems: Human Cognitive Models in Systems Design.* Hillsdale, NJ: Lawrence Erlbaum Associates, 67-89.

22. McCrickard, D. S., Chewar, C. M., Somervell, J. P., and Ndiwalana, A. (2003). A Model for Notification Systems Evaluation—Assessing User Goals for Multitasking Activity. *ACM Transactions on Computer-Human Interaction (TOCHI)*, Vol. 10, Issue 4, 312-338.

23. Newman, W. M. (1997). Better or just different? On the benefits of designing interactive systems in terms of critical parameters. *Proc. DIS 1997*, 239-245.

24. Nielsen, J. (1993). Usability Engineering. Morgan Kaufman, San Diego, CA.

25. Norman, D. A. (1986). Cognitive engineering. In *Norman, D. A., and Draper, S. W. (Eds.), User Centered System Design*, Hillsdale, NJ: Lawrence Erlbaum Associates, 31-62.

26. Payne, C., Allgood, C. F., Chewar, C. M., Holbrook, C., and McCrickard, D. S. (2003). Generalizing Interface Design Knowledge: Lessons Learned from Developing a Claims Library. *Proc. IRI 2003*, 362-369.

27. Rosson, M. B. and Carroll, J. M. (2002). Usability Engineering: Scenario-Based Development of Human-Computer Interaction. Morgan Kaufman, New York, NY.

28. Sutcliffe, A. G. (2000). On the Effective Use and Reuse of HCI Knowledge. *ACM Transactions on Computer-Human Interaction (TOCHI)*, Vol. 7, Issue 2, 197-221.

29. Sutcliffe, A. G. and Carroll, J. M. (1999). Designing Claims for Reuse in Interactive Systems Design. *International Journal of Human-Computer Studies*, Vol. 50, Issue 3, 213-241.

30. Wahid, S., and McCrickard, D. S. (2006). Claims Maps: Treasure Maps for Scenario-Based Design. *Proc. ED-MEDIA 2006*, 553-560.

31. Wahid, S., Smith, J. L., Berry, B., Chewar, C. M., and McCrickard, D. S. (2004). Visualization of Design Knowledge Component Relationships to Facilitate Reuse. *Proc. IRI 2004*, 414-419.

32. Winograd, T. (1995). From Programming Environments to Environments for Designing. In *Communications of the ACM (CACM)*. Vol. 38, Issue 6, 65-74.

33. Wixon, D. (2003). Evaluating usability methods: why the current literature fails the practitioner. *interactions*, Vol. 10, no. 4, 28-34.