Spiraling Toward Usability: An Integrated Design Environment and Management System

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

Center for Human-Computer Interaction[1]
Department of Computer Science
Virginia Polytechnic Institute & State University
Blacksburg, VA 24060-0106
{chonglee, swahid, bconglet, mccricks}@cs.vt.edu

Department of Electrical Engineering and Computer Science[2]
United States Military Academy
West Point, NY 10996
christa.chewar@usma.edu

ABSTRACT

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 introduces an integrated design environment and knowledge management system, LINK-UP. The central design record (CDR) module, provides tools to enable a guided, coherent development process. The CDR aims to prevent breakdowns occurring between design and evaluation phases—both within the development team and during design knowledge reuse processes. We report on results from three case studies illustrating novice designers' use of LINK-UP. 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. Although LINK-UP supports a specific design domain, our IDE approach can transfer to other domains.

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

INTRODUCTION

Computing systems are becoming increasingly pervasive and continue to affect and mediate our daily lives in new ways. Many different usability engineering methodologies and techniques aim to bring some structure to the design process and allow developers to develop these 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 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 our 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 [3, 14], this paper introduces the central design record (CDR), a design representation that makes explicit where and how handoffs occur in the development process and highlight design decisions that need reconsideration during the next development iteration. For instance, the CDR allows developers to determine links between current practice and how and why an interface modifies it or relationships between evaluation data and different aspects of the interface.

This paper documents a series of three case studies that illustrate how novice designers used LINK-UP to engineer interfaces. Building on our experiences with knowledge repositories and principled process-oriented development, LINK-UP serves as a culmination and realization of our previous work. Our results suggest that a design knowledge IDE that incorporates the processes and principles of the CDR, can help novice developers apply a design methodology to develop interfaces in a guided, methodical fashion and avoid process breakdowns. Furthermore, we see evidence that, through the use of the CDR, iterative application and verification of reusable design knowledge is made practical for novice interface developers.

MOTIVATION

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

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 [13]. 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. 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. *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 [21]. 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 [11]. 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 [12]. Numerous approaches to usability engineering have been proposed and used that share many similar characteristics such as user-centeredness and iterative design coupled with analytic and empirical evaluations.

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 [2, 15]. *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[6, 9, 17]. 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 [17].

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 have no easy way to discuss and reflect on designs. Borchers advocated the use of formally defined patterns to support communication among stakeholders [1]. 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 [1, 5]. Sutcliffe pointed to structured claims as a way to delivering HCI research knowledge to practitioners, giving them the ability to communicate through claims [16].

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 [20]. Such systems can allow developers to better use and integrate usability design processes such as those developed by Rosson and Carroll.

Key focus points for Design Knowledge IDEs

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

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.

LINK-UP

To address the concern of providing tool support to designers and achieve the vision we described in the previous section, we 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. We continue our introduction of LINK-UP 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.

Notification Systems

A focus on a particular interface design domain allows us to increase our depth rather than focusing on the breadth of a larger domain, an important concern when dealing with design knowledge. In our case, the LINK-UP system focuses on the notification systems domain, a growing class of applications supporting information needs. In such systems a user acts within a primary task while explicitly or implicitly monitoring information through a notification system as a secondary task. 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 [9]. Instant messengers and e-mail alerts are common examples of such systems. Ambient displays, large screen information exhibits, and car navigation systems are other examples of off-the-desktop 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. 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 [9]. The framework uses IRC values ranging from 0 to 1 for every critical parameter. Each integer value combination describes one of eight possible notification systems. Designers can use these critical parameters as targets for their notification system as well as usability and performance metrics for comparison during evaluations.

LINK-UP Development Process

With the four focus points in mind, we aimed to create a system that will support the design of notification systems through the use of IRC and SBD's notion of claims. LINK-UP consists of a design knowledge repository and modules in which design activities are carried out. The repository, or the *claims library* [14], contains claims related to our domain. Designers can search for reusable claims applicable in their own designs by using various searching or browsing [18] features. This basis, a structured collection of claims, supports knowledge sharing among designer communities interested in the domain. Figure 1 shows the key knowledge structures and cyclical iterative approach designers use in creating interfaces. The remainder of this section describes the structure and approach.

The basic structure of a claim consists of a feature and a list of upside and downside tradeoffs. We extend this structure with additional information. 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 describes a task in which the claim can come into use. The scenario allows designers to consider how the claim can be used within the context of a design. 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.

The contents of the library are finite. 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 process. 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 of the LINK-UP environment. Upon it lie two different modules: the requirements analysis module and the central design record module [7]. The requirements analysis module, where the designer starts, is an environment in which the designer determines design goals and establishes the problems that must be solved. The CDR module is the environment in which much of the design is created. Both modules support the reuse and creation of claims.

Within the requirements analysis module, the process of SBD 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.

To help clarify the goal of a new design, based on the problem scenario, the designer determines a target IRC value they would like to achieve through their design. This situates the new system within the general design space for the design domain (notification systems). Furthermore, the processes serve as a formalization of their goals and provides 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 [3] that asks various questions regarding the nature of the system they wish to design.

As a step toward creating a detailed design representation, 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 [13] and place concepts within each stage. Such decompositions help designers understand how users interact with the existing system 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.

Once the requirements analysis is completed, the designer moves on to the *central design record module*. The CDR module is designed to support the activity, information, and interaction phases of SBD. When a user enters the module, the problem scenario and claims from the requirements analysis module are imported.

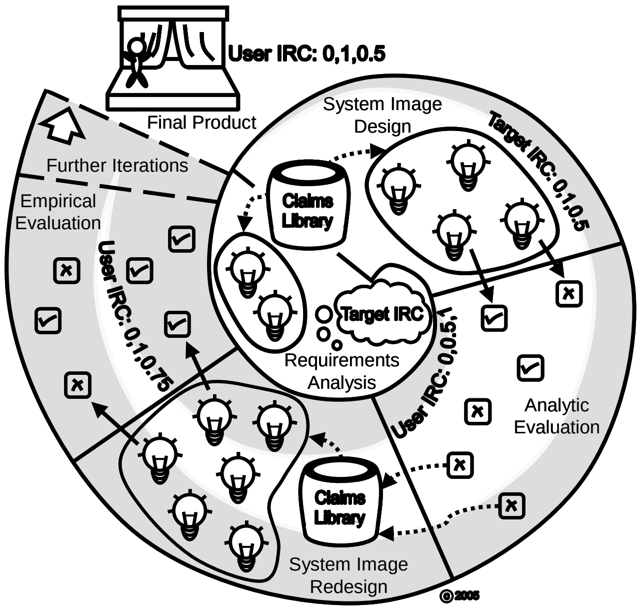

Figure 1. LINK-UP's knowledge structures and design processes. Designers start at the center identifying requirements and a target IRC goal representation. Design iterations result in CDRs that include possible design claims, which are tested through an evaluation, and lead to eventual convergence of the design model and user's model.

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. Once the scenario is written, designers begin a claims analysis process for the scenario in which they gather claims for the activity scenario. 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 nature of the CDR module is very fluid. A specific process, as opposed to the requirements analysis module, is not imposed upon the designer. 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 contribute to a detailed representation of a design that can be used for evaluation purposes. For example, the breakdown of tasks in terms of 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.

CASE STUDIES

In this section we present three cases studies of design projects developed using LINK-UP over several development iterations. 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 was chosen a general theme for all the projects. Each project group consisted of 4-5 students. Three exemplar systems were chosen among the groups in the class based on the quality of the feedback they provided in a series of project reports 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. 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 to facilitate general enjoyment of attractions on and around the Huckleberry Trail, an outdoor 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 study illustrates how the CDR can support principled incremental interface improvement by linking evaluation results directly to design decisions.

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. 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's surroundings without disrupting enjoyment.

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. The developers found the overall process 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. 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 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. *This illustrates the link between design rationale in the CDR to empirical results and to the system* *goal (defined in terms of IRC)* made possible by the persistent representation of the design model that is stored and maintained through the CDR module.

The case highlights the general success of using critical parameters to measure the success of the system at achieving initial goals while using the design model stored in LINK-UP to direct iterative refinements to achieve them. We expect that similar successes would become apparent through further iterations, more refined prototypes and testing on the actual 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, the location of the alert, and the best route to the alert site. This is meant to mitigate the inefficiency of current radio-based information relaying between officers. 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.

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 of their system and determined that it should have high interruption to alert officers of 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 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 ultimately decided not to mitigate this problem because they believed the 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 participants were undergraduate students with several years of driving experience and average experience using online map services. The designed tests were derived from design claims in the CDR and were primarily focused on how effectively the system could notify drivers without too much interruption and whether drivers were able to comprehend the information provided. The results of their study indicated that their audio notifications seemed to be too interruptive as they caused participants to crash their vehicle. In addition, the participants found it too difficult to follow and understand the map. 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*. 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. 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.

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*

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	(+) System image 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]*	(+) System image 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]*	→	(+) 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.

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 act as a *common language* through which project stakeholders can both discuss the design and resolve any misunderstandings [1].

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. 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. 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.

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.

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.

DISCUSSION

In this section, we synthesize the results from the case studies and draw out conclusions about LINK-UP. We discuss both the strengths of our approach and areas for improvement, summarized in Table 1, with respect to the four focus points. These highlight the value of our approach and 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 value of integrating critical parameters throughout the development process. 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. 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 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 encapsulated 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 was helpful to novice developers, particularly in early stages of design, in identifying problems and claims. However, all groups found the 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 [4]. 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. 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 design representation 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.

Overall, the LINK-UP system did guide usability engineering processes and support knowledge creation while sacrificing time-efficiency and flexibility in the design process. We do not feel that these are critical drawbacks, since LINK-UP is intended for novice developers. As LINK-UP's users become more accustomed to a design process, they can relax process requirements in favor of efficiency and flexibility. 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.

CONCLUSIONS AND FUTURE WORK

The ever increasing pervasiveness of computing systems in and around our lives is mirrored by the development of numerous usability methodologies and techniques to aid in their development. However, it is not always clear to novice developers how to use these different processes and techniques in a coherent design process. We demonstrate through the case studies that a design knowledge IDE, centered on the central design record, can help 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. We also show that the use of CDR supports the application and verification of reusable design knowledge for novice developers.

Based on our results, we derive the following guidelines for design knowledge integrated development environments that incorporate a module patterned on our CDR:

- Persistent design representations should support multiple or 'current' perspectives to direct development efforts on salient design concerns.
- Design rationale should be tightly coupled to evaluation data 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 stakeholder collaborations.
- Design knowledge IDEs should support, but not require, guided processes to aid in knowledge capture, knowledge use and goal formation.

The current iteration of LINK-UP was developed to support novice developers. There are tradeoffs inherent in developing a design knowledge IDE to support these types of developers. Professional designers would likely find the current implementation too constraining to apply in an industrial setting. Future efforts will address the needs of professional interface and system developers so they can bring reusable design knowledge to bear in practical design projects. Nonetheless, the current implementation demonstrates the potential for a reuse-enabled integrated design environment in guiding novice developers through the interface design and evaluation process. Our case studies, combined with the resulting guidelines can 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. We hope that our work motivates additional efforts to develop a principled, scientific approach to interaction design.

ACKNOWLEDGMENTS

We wish to thank Edwin Bachetti and the other LINK-UP developers, and the students in the CS 3724 Human-Computer Interaction course.

REFERENCES

1. Borchers, J. O. A Pattern Approach to Interaction Design. *Proc. DIS 2000*, (2000) 369-378.
2. Carroll, J. M. and Kellogg, W. A. Artifact as theory-nexus: Hermeneutics meets theory-based design. *Proc. CHI 1989*, (1989), 7-14.
3. Chewar, C. M., Bachetti, E., McCrickard D. S., and Booker, J. Automating a Design Reuse Facility with Critical Parameters: Lessons Learned in Developing the LINK-UP System." *Proc. CADUI 2004*, (2004) 236-247.
4. Chewar, C. M., and McCrickard, D. S. Links for a Human-Centered Science of Design: Integrated Design Knowledge Environments for a Software Development Process. *Proc. HICSS 2005*, (2005), 10 pgs (CD-ROM).
5. Erickson, T. Lingua Francas for Design: Sacred Places and Pattern Languages. *Proc. DIS 2000*, (2000), 357-368
6. Gamma, E., Helm, R., Johnson, R., and Vlissides, J. *Design Patterns: elements of reusable object-oriented software*. Boston: Addison-Wesley Longman. 1995
7. Lee, J. C., Chewar, C. M., and McCrickard, D. S. Image is Everything: Advancing HCI Knowledge and Interface Design Using the System Image. *Proc. ACMSE 2005*, (2005), Vol. 2, 376-381.
8. Lee, J. C., Lin, S., Chewar, C. M., McCrickard, D. S., Fabian, A., and Jackson, A. From Chaos to Cooperation: Teaching Analytic Evaluation with LINK-UP. *Proc. E-Learn 2004*, (2004), 2755-2762.
9. Majchrzak, L., Cooper, L. P., and Neece, O. E. Knowledge Reuse for Innovation. *Management Science 50*(2): 174-188, Feb 2004.
10. McCrickard, D. S., Chewar, C. M., Somervell, J. P., and Ndiwalana, A. A Model for Notification Systems Evaluation—Assessing User Goals for Multitasking Activity. *ACM Transactions on Computer-Human Interaction (TOCHI)*, Dec ’03, Vol. 10, Issue 4, 312-338
11. Newman, W. M. Better or just different? On the benefits of designing interactive systems in terms of critical parameters. *Proc. DIS 1997*, (1997) 239-245
12. Nielsen, J. Usability Engineering. Morgan Kaufman, San Diego, CA 1993.
13. Norman, D. A. Cognitive engineering. In *D. A. Norman & S. W. Draper, Eds. User Centered System Design*, Hillsdale, NJ: Erlbaum, 31-62
14. Payne, C., Allgood, C. F., Chewar, C. M., Holbrook, C., and McCrickard, D. S. Generalizing Interface Design Knowledge: Lessons Learned from Developing a Claims Library. *Proc. IRI 2003*, (2003), 362-369.
15. Rosson, M. B. and Carroll, J. M. Usability Engineering: Scenario-Based Development of Human-Computer Interaction. Morgan Kaufman, New York, NY, 2002.
16. Sutcliffe, A. G. On the Effective Use and Reuse of HCI Knowledge. *ACM Transactions on Computer-Human Interaction (TOCHI)*, Jun 2000, Vol. 7, Issue 2, 197-221
17. Sutcliffe, A. G. and Carroll, J. M. Designing Claims for Reuse in Interactive Systems Design. *International Journal of Human-Computer Studies*, Vol. 50, Issue 3, 213-241
18. Wahid, S., Smith, J. L., Berry, B., Chewar, C. M., and McCrickard, D. S. Visualization of Design Knowledge Component Relationships to Facilitate Reuse. In *Proc. IRI 2004*, (2004), 414-419.
19. Whittaker, S., Terveen, L., and Nardi, B. A. Let's stop pushing the envelope and start addressing it: A reference task agenda for HCI. *Human-Computer Interaction*, 15, 75-106.
20. Winograd, T. From Programming Environments to Environments for Designing. In *Communications of the ACM (CACM)*. June 1995, Vol. 38, Issue 6, 65-74
21. Wixon, D. Evaluating usability methods: why the current literature fails the practitioner. *interactions*, Vol. 10, no. 4, July+August 2003.