Supporting Collaborative Design through Risk Analysis

Benefits of Calculated Risk in the Design of Interactive Systems

Jamie Laine Smith

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

D. Scott McCrickard, Chair
Shawn Bohner
Linda Wallace

17 May 2005
Blacksburg, Virginia

Keywords: risk management, scenario-based design, claims, notification systems, human-computer interaction, project management

Supporting Collaborative Design through Risk Analysis

Benefits of Calculated Risk in the Design of Interactive Systems

Jamie Laine Smith

Abstract

As software systems continue to grow, and as project teams become larger and more distributed, support for project management in collaborative environments is critical. Management tasks include maintaining team coordination, monitoring progress, and, of particular interest for this work, managing risk tasks often add significant overhead to a project. To reduce overhead, management tasks must be integrated, whenever possible, directly into the software design and development process. Additionally, to prevent common problems from recurring in different projects, developers must focus on reusing the knowledge gained and the lessons learned through previous projects to guide future endeavors.

The overall goals of the work contained within this thesis are to define reusable, project-related knowledge as project risks and to utilize that knowledge in the development of a risk-driven management model to be integrated within a human-computer interaction (HCI) design process. Existing risk management techniques typically involve process-related knowledge, such as project planning and client involvement. However, HCI as a discipline is more concerned with product-related design knowledge. Claims structure product-related knowledge for reuse by explicitly stating the positive and negative tradeoffs of incorporating a particular feature in the design of a system. By managing these negative tradeoffs as design risks, HCI designers can identify and focus on the most critical design issues throughout the course of a project. This systematic approach to solving design issues helps to ensure that designers make informed design decisions rather than following an *ad hoc* design process.

Building upon existing risk management techniques from other domains, this thesis delivers a risk-driven, claims-based management model for HCI design. In doing so, this work transfers techniques traditionally used in managing process-related knowledge into a new domain for use in managing product-related design risks. The need for risk management in software design is argued through a review of existing collaborative tools, resulting in a series of guidelines for providing project management support. An initial risk model is then presented, along with the results of a user evaluation conducted to determine not only the accuracy of risk prioritization, but also the overall benefit of applying risk management within the context of HCI design. Following a discussion of these results, several directions for future work are mentioned both to further the quest for a true design science and to improve the standards by which software projects are managed.

"Take calculated risks. That is quite different from being rash."

-- George S. Patton, US general

Acknowledgements

I am who I am in large part because of the many wonderful people in my life. I have a number of individuals to thank for helping me to achieve this goal and for making my time at Virginia Tech so memorable.

First and foremost, to my parents…you have stood behind me from day one, always encouraging me to take the next step and follow my dreams. You allowed me to pursue my own path, you supported me through all my bad days, and you commended me when I finally succeeded. I love you, and I thank you, and I hope I've made you proud.

To Ashley and Bob…as I think back over the past two years, I can pinpoint only a handful of occasions that we were able to spend together. However, many of my fondest memories of this time have been spent in your company. Thank you for your love and support and for the endless good times. Ash, after all this time, we have finally managed to synchronize our graduations, which means that we can celebrate together. Congratulations to an amazing sister and friend.

To Dr. McCrickard…spending the past two years under your advisement has made graduate school far more enjoyable than I ever imagined it could be. There were times when I thought I might never see this day, but you always knew when and how to push me and how to keep me smiling in the process. Maybe I will be back one day…but only if you promise me a bigger desk.

To Dr. Bohner and Dr. Wallace…thank you for joining my committee and for supporting my work. I appreciate your patience and your feedback.

To Goldie…thank you for taking the time to participate in my evaluation. Your feedback has been invaluable.

To Shahtab, Jason, and Ali…confined spaces strengthen friendships, and I can't imagine three guys with which I would have rather spent the majority of the past two years. Thank you for the good times…the endless banter, the frequent lunches spent contemplating the ways of the world, and, of course, your constructive comments on my work.

To Edwin and your extraordinary crew of developers…I probably would not be where I am today without your hard work and countless late nights spent coding in the lab. Edwin, thank you for making the final months of my graduate career far more enjoyable than they should have been.

To Shan…through endless emails and instant messages, frequent phone calls, and the occasional visit, we have managed to remain friends from different coasts. You know me better than anyone else on this list. We have been through a lot together, and through it all you have stood by me, picked me up on numerous occasions, made me laugh, and reminded me of my worth. Thank you for being an amazing friend. Thank you for loving me for me.

And finally, to all of my family and friends…I appreciate your love and support now and always

Contents

List of Figures

List of Tables

Chapter 1 : Introduction

The notion of risk dates back to ancient Greek and Egyptian civilizations. From as early as 3500 BC, Egyptian tomb paintings depicted the use of primitive dice in an ancient game of chance. Greek mythology explains the creation of the world as a contest among the gods, in which Zeus won the heavens with a lucky roll of the dice. This convention translated to daily life in ancient Greece, as fate was relinquished to the will of the gods. Worship and sacrifice were considered the only manner of shifting one's fate [9].

The Greeks, as masters of logic and theory, gave considerable thought to the notion of probability. In fact, Socrates defines *eikos*, meaning probable [9], as "likeness to truth." This ancient characterization is surprisingly similar to the modern definition of probability, i.e., the "likelihood that a given event will occur" [2]. However, likeness to truth is not the same as truth, and the Greeks insisted on proof through logic or other truisms. Lacking an adequate number system to allow the calculation of probability, they chose, instead, to increase their luck by appeasing the gods. It was not until the Renaissance that man began to question his luck at gambling and establish the basic rules of probability. Once man determined that he could take control of his own future, he began to understand the concept of risk, measure it, and base his decisions on the weight of its consequences.

Based on this account of the history of probability, Peter Bernstein, in *Against the Gods* [9], argues that our modern understanding of the concept of risk has not only been developing for centuries, but also is a key factor in distinguishing ancient history from modern civilization. Since the development of mathematical principles for determining probability, risk management has played a critical role not only in strategizing a game of chance, but more importantly, in developing modern infrastructure –from building bridges to producing electrical power, from preventing polio to achieving flight. Most everyone possesses a similar understanding of the concept of risk; however, the challenge is in determining how best to describe, measure, and communicate risk. The increase in interaction among critical infrastructure in different fields has made risk management a complex problem for modern times.

In the past, structural failures were frequent yet isolated. Today, structural complexity has increased to include large-scale, software-intensive systems, and failures remain frequent. Moreover, the vulnerability of complex technological systems is inherently linked to that of nearly all critical infrastructures, from banking to communication to national security. When a system failure occurs in one domain, it can have disastrous effects across multiple other domains. To reduce the possibility of system failure, software professionals must adopt adequate techniques for identifying and assessing project-related risks. As system size and complexity continues to increase, this task will become more difficult, yet more critical, to accomplish. Based on his years of experience managing large, complex combat systems, Rear Admiral Bill Carlson, United States Navy, supports this claim, stating that, "if you're not managing risk, you're managing the wrong thing" [11].

1.1 Project Management in the Software Domain

Project management is a relatively immature consideration within the domain of software development. Management techniques, adopted from the business world, provide guidance to a software project manager; however, they add substantial overhead to projects in a field notorious for falling behind schedule. A rapid increase in the size and distribution of project teams further complicates the management issue by disrupting the traditional group dynamics upon which many management paradigms are based. Consequently, software developers must devise new methods of maintaining team coordination, monitoring progress, and managing risk.

Inadequate project management leads to the failure of many software projects. In 1995, Capers Jones described the distribution of software projects based on their size and point of delivery [31]. Small-scale projects, estimated between one and ten function points, were delivered on time 83.16 percent of the time, with only 1.92 percent being delayed and 0.25 percent being canceled. In contrast, only 13.67 percent of large-scale projects, estimated at 100,000 function points, were delivered on time, 21.33 percent were delayed, and 65.00 percent were canceled. These statistics are supported by The Standish Group's [52] 1994 CHAOS Report, which declared a 16 percent project success rate and a 31 percent project failure rate. The remaining 53 percent of projects were completed but challenged by the customer. The 2003 CHAOS report depicted significant improvements within the software domain, reporting a 34 percent project success rate and a 15 percent project failure rate. However, the remaining 51 percent of projects remain challenged, often falling victim to serious cost and schedule overruns. Additionally, the 2003 report shows that, on average, only 52 percent of required features and functionality are actually implemented in delivered systems, a decline from 67 percent in 2000. The 2003 report surveyed 13,522 software projects.

Although these statistics show improvement in some areas of software project management, the overall project success rate remains extremely low. Very few projects are delivered on time and within budget, and those projects that are delivered successfully rarely satisfy more than half of the system requirements. To increase project success rates on a large scale, software professionals must not only improve their management techniques, but also change the way they think about project management. They must stop thinking of management tasks as expendable overhead and start to integrate them directly within their design and development processes. This thesis seeks to facilitate that integration by injecting risk management directly in the software design process. As a result, management tasks can begin to occur naturally as a byproduct of following a systematic approach to design.

1.2 Collaborative Design of Complex Systems

The design of software-intensive systems is becoming an increasingly complex task. This complexity results from the need to integrate different views of a problem, manage large amounts of project-related knowledge, and understand the rationale behind decisions as the design evolves throughout the life of a project. A successful system design requires a "cross-section" of knowledge, skills, and alternative perspectives [8]. However, system complexity makes it difficult for any individual to achieve a full understanding of all aspects of a design. The breadth of knowledge needed for the design of a large-scale system is usually distributed among a team of individuals who can learn from one another through shared knowledge and understanding [4]. Globalization has popularized the use of distributed project teams, and this increase in team distribution further complicates both collaborative design and

project management. Project teams exhibit varying degrees of distribution, from across campus to across the globe, and each team requires a custom set of tools to enable effective collaboration. These tools include not only appropriate paradigms for the management of complex, collaborative projects, but also supportive technologies to facilitate the use of such techniques.

Although hurdles exist at all stages of the development life-cycle, focusing on the effective design of complex systems can help to diminish subsequent problems during implementation, assessment, deployment, and maintenance. Human-computer interaction (HCI) is concerned with designing interfaces to interactive systems that allow users to accomplish their goals. A key aim of HCI is to inject knowledge from psychology, sociology, and other relevant disciplines into the process of designing user interfaces so that usability problems can be detected and diagnosed early. Successful design increasingly requires a well-constructed and well-managed team that follows a systematic approach and designs with the needs of all project stakeholders in mind. Although researchers and practitioners are making strides toward this design science, substantial questions remain:

- How do individuals collaborate on a design project? What knowledge do designers from different perspectives bring to the table, and how can that knowledge be externalized, communicated, and stored for use by everyone on the team?
- How can project teams coordinate their knowledge and their actions to complete projects successfully? What management tasks are most critical, and how can those tasks be accomplished without squandering valuable time and effort?
- How can the process of applying design theory to practice be transformed into a true scientific method? What knowledge can be collected throughout the design process and utilized in managing a design project and improving a design artifact?
- Why should all design projects begin from scratch? How can knowledge obtained about the design product and the design process be captured for reuse in subsequent projects?

These questions span an array of issues within HCI design, software engineering, and project management. To arrive at an appropriate science of design, researchers must first make individual strides in various directions, incorporating knowledge and expertise from multiple disciplines into a formal management paradigm. Given the complexity of design, teams that are managed properly and that follow a systematic design approach are more likely to produce a quality product. This thesis takes a step toward these high-level design and management goals by integrating risk management techniques within an HCI design process.

1.3 Integrating Management and Design for Mutual Benefit

The concept of risk implies the possibility of an event with negative consequences. Risk management involves identifying risks and resolving the most critical issues based on an estimation of the potential impact of the risk and its probability of occurring. Existing risk management techniques allow project managers to make valuable risk predictions and, consequently, avoid project failure. However, these techniques, particularly within the software

domain, primarily manage process-related issues, such as project scheduling, budget, and customer involvement. Although such issues are critical to the success of all types of projects, HCI as a discipline is concerned with another category of project risks. Product-related risks threaten the success of the design artifact, i.e., the system interface. One goal of HCI design is to identify and resolve potential problems within an interface that might prevent users from accomplishing their goals. Unfortunately, the current process of identifying and resolving HCI design issues is *ad hoc* at best. The work contained within this thesis adds structure to the design process by facilitating the identification and resolution of key design issues, as stated below.

> *Building upon prior success of process-related risk management techniques, this thesis institutes a risk-driven management model suitable for assessing the criticality of product-related design concerns within a particular context of use –leading to a thorough assessment of design risk, increased focus on critical design issues, and support for design discussion and knowledge reuse.*

The remainder of this document elaborates on the integration of a risk-driven management model within a scenario-based HCI design process, both to further the development of a systematic design approach and to improve the standards by which software-intensive projects are managed.

Given the increase in system size and complexity, designers cannot expect to resolve all possible design concerns. Through use of a risk model, designers prioritize product-related risks, improving design efficiency by identifying key concerns more quickly and focusing on the resolution of those issues. By resolving critical usability issues, designers increase the probability of achieving design goals and, consequently, user satisfaction. Risk estimates in other domains typically rely on the utilization of all available information to make the most accurate prediction possible. Likewise, an assessment of design concerns should also take advantage of all available information. Decisions regarding modifications to a system design should be guided by a complete appraisal of knowledge from previous projects, information illustrated by system requirements, and feedback from end-users and other project stakeholders. Analysis of product-related design risks formalizes the design process by drawing attention to key problems in an existing design and by helping project teams to remain focused on the most critical issues for improvement at any stage of the design process.

While improving their design process and the quality of the resulting artifact, designers also improve project management through use of this risk model. Dynamic prioritization of design risks, beginning early in the design phase and continuing throughout the project, allows a team of designers to monitor changes in risk priority as their design evolves over time. Additionally, as team members work to resolve design issues, use of the risk model facilitates task allocation and monitoring of progress. Implementation of the risk model within a collaborative design environment can also improve awareness among team members in distributed situations. Moreover, by incorporating the risk model into the design process, risk management is accomplished with minimal added overhead.

The design and management issues discussed above are detailed in the following chapters. Chapter 2 provides the background needed to understand the work contained within this thesis. Focusing first on HCI design, the chapter discusses in detail the foundations of Scenario-Based Design (SBD), including the concepts of scenarios and claims and the key steps in the SBD process. For the purposes of this work, SBD is applied to the design of notification systems, which grant users access to information outside the scope of their primary task. A web-based design environment, called LINK-UP, supports the SBD process and also promotes reuse of design knowledge within this domain. The remainder of Chapter 2 introduces software project management and outlines the strengths and weaknesses of current management techniques, based on a review of several existing tool technologies. This chapter also introduces the foundations of risk analysis and explains the shortcomings of existing techniques for the management of collaborative software design teams.

Chapter 3 expands upon the concepts of risk, showing how risk management can be incorporated into the SBD process. This chapter discusses the differences between product and process-related knowledge and how both forms of knowledge might be stored and reused to aid in the management of software projects. Key components for analyzing product-related design risk are described. The chapter outlines the overall contributions of this work, showing how risk management can improve the SBD process and how components of SBD facilitate risk management.

Chapter 4 details a risk-driven, claims-centric management model, applied to SBD and the design of notification systems. This chapter provides a set of criteria and equations for calculating and prioritizing product-related design risks, representing an initial, proof of concept model for use in evaluating the benefit of risk management within the context of HCI design. The implementation of this risk model within the LINK-UP system is described, and the flexibility, scalability, and limitations of the initial model are discussed.

Chapter 5 presents the experimental design and results of an initial user evaluation of the risk model. The chapter begins with a case study involving the redesign of an existing notification system. A lead designer on the project was asked to assess the risk model in relation to the existing system design and comment on its accuracy and usefulness. The chapter then presents the details of a more in-depth user evaluation of the risk model. Student design teams, following the SBD process and using LINK-UP to manage design knowledge for a semester project, were asked to follow the risk analysis steps in conjunction with their design sequence. They were then asked to examine and assess the risk model and use its results to guide the redesign phase of their project. Feedback from this evaluation was then used to assess the benefit of risk management to HCI design and determine the most important factors to consider in estimating design risk.

Chapter 6 concludes this thesis with a discussion of the results of the user evaluations and speculation about improvements to the initial risk model. This chapter returns to a discussion of the overall goals of the HCI design and project management, restating the contributions of this research and defining critical directions for future work

Chapter 2 : Related Work

The underlying research outlined in this chapter lays a foundation for the work contained within this thesis. To assess the need for management support in software design projects, it is first necessary to understand the strengths and weaknesses of the HCI design process. Although many design methodologies exist, this work focuses on the use of Scenario-Based Design (SBD), which is discussed in Section 2.1. SBD facilitates design discussion among and between technical designers and non-technical project stakeholders by conveying design knowledge through natural language scenarios. Additionally, SBD extracts key design features and their positive and negative consequences in the form of claims. These consequences represent design opportunities as well as design risks; thus, SBD facilitates the integration of a risk model within the design process.

Design is a creative process; however, system design almost always builds upon knowledge acquired during previous design endeavors. Section 2.2 outlines the contributions of SBD toward the reuse of design knowledge. An extension of code reuse in software development, design knowledge reuse improves project efficiency by providing a foundation of previous work on which to build. Additionally, knowledge reuse ensures that successful ideas are remembered and common mistakes are avoided. To limit the scope of reuse, this work focuses on the application and reuse of knowledge in the design of notification systems, which provide users access to information outside the scope of their primary task. Notification systems span a broad range of applications, creating a focused yet copious research domain. Several tool technologies, including a knowledge repository and associated collaborative design environment, are being developed to support SBD and to facilitate design knowledge reuse within the domain of notification systems. To be successful, these tools must support effective teamwork and project management in addition to sound HCI design. These tools, which are also described in Section 2.2, will serve as a test bed for integration of a risk model into the SBD process.

Goals of this work include not only furthering the science of design, but also improving management techniques in software design projects through the integration of risk analysis. Section 2.3 discusses project management within the context of software and HCI design. This section discusses changing team dynamics with respect to both design and software-intensive projects and outlines the strengths and weaknesses of existing project management tool technologies, articulating the need for new management paradigms. The section then introduces the concept and importance of risk management within the software domain, expressing strengths and weaknesses of existing risk management techniques and outlining key risk components that contribute to the development of a risk-driven model for HCI design.

2.1 Scenario-Based Design

As software-intensive systems continue to grow in size and complexity, their utility focuses less on functionality and more on the ability to transform human activity. As computers continue to be incorporated into all

aspects of our lives, from education to the workplace, from business to entertainment, we rely not only on their ability to accomplish tasks but also on our ability to interact with them to achieve our goals. In this age of technology, humans must learn to live with computers. Likewise, computers must "learn to live" with humans, or rather, software designers must facilitate this cohabitation by developing systems that meet the needs of their users.

The design of useful and usable systems relies on the ability of designers, end-users, and other system stakeholders to communicate with one another about user-system interaction. End-users must successfully convey their needs to designers, who must then explain to users how they can interact with a system to achieve those goals. To create a satisfactory system, designers and end-users must share this bidirectional view of system usage. Designers often document their ideas as functional specifications, which can be difficult for end-users, other system stakeholders, and even other designers, to understand. These individuals can communicate their views of system usage more effectively with natural language descriptions, termed usage scenarios. These simple, narrative descriptions of existing or envisioned system use are the foundation of scenario-based design [12].

2.1.1 Toward a Science of Design

As system complexity increases, so does the complexity of system interfaces. Successful design increasingly requires a well-constructed, well-trained, and well-managed team that follows a systematic approach to applying scientific knowledge to design practice. Advancement toward a true science of design involves the quest for an "intellectually rigorous, formalized, and teachable body of knowledge about the principles underlying software-intensive systems and the processes used to create them" [23]. Drawing from the work of Simon [50], a true science of design must involve identifying and selecting design alternatives, understanding the differences between those alternatives as well as the validity of each, and ensuring that design failures are not repeated. At the same time, a scientific approach to design must preserve the creativity, style, and innovation associated with the design process.

Carroll begins to bridge the gap between software design and HCI by stating that design must thoroughly incorporate the impact of interactive systems on end-users and other system stakeholders [15]. In this way, effective design must involve understanding people and their activities and incorporating new designs into existing human activity systems. In this way, scenario-based design represents the beginning of a systematic approach that accounts for a broader design perspective. To be successful, this "engineering approach" to HCI must complement current software engineering paradigms yet involve the analysis of design rationale to ensure that socio-technical systems are designed with the user in mind [13, 54].

Scenario-based design is motivated by the goal of transforming HCI into a scientific domain. It attempts to structure the design process by guiding designers through analysis of end-user tasks, consideration of design tradeoffs, and decision-making based on sound rationale. Scenarios provide concise and descriptive accounts of system use by showing what tasks users attempt to perform, what problems they encounter, and how they interpret success or failure. Thus, a comprehensive set of scenarios represents a solid task analysis for the system being developed [12]. Following the scenario creation process, designers can extract key issues from each scenario and explicitly state design tradeoffs in the form of claims. The claims analysis process is discussed in detail in the next section.

2.1.2 Claims Analysis

A comprehensive set of scenarios provides a strong task analysis for a system; however, scenarios do not explicitly state the underlying causal relationships inherent in the design. To better manage and understand HCI concepts and principles, designers must focus on relevant artifact features and explicit design tradeoffs [14]. Claims are one form proposed for the expression of HCI design knowledge associated with a particular context of use.

2.1.2.1 Achieving Task Coverage

The goal of the claims analysis process is to produce a set of claims with sufficient task coverage for the design of a system, where task coverage refers to the degree to which a particular system design contains claims for each of Norman's stages of action.

Norman's theory of action [39] recognizes that the usability of an interactive system relies on the convergence of two conceptual models –the design model, which accounts for the designers' goals for the system, and the user model, which accounts for the end-user's actual understanding of and experience with the system. The two models must come together, creating a vision, or *system image*, that is shared by both designer and end-user.

Each of these models can be analyzed as stages of action, which depict the cycle of task evaluation and task execution across two gulfs, namely, the *Gulf of Execution* and the *Gulf of Evaluation*. To accomplish a goal, the completion of two key tasks is necessary. Based on the goal, a specific action is taken to manipulate the world –i.e., a plan of action is *executed*. Then, the results of that action are examined to determine if the goal was accomplished –i.e., the plan of action is *evaluated*. The Gulf of Execution contains three stages of action –forming the intention, specifying an action sequence, and executing an action. The Gulf of Evaluation also contains three stages of action – perceiving the state of the world, interpreting the state of the world, and evaluating, or making sense, of the outcome. The inclusion of claims in each stage of action ensures that the designer has considered each step in the user's process of accomplishing a particular goal and that all necessary user tasks have been covered in the design.

2.1.2.2 Structuring a Claim

Through claims analysis, designers extract into a compact and understandable form the key issues contained within a scenario. Claims can cover a wide range of design problems and principles by making explicit the upside and downside tradeoffs implicitly narrated in a usage scenario [14].

Like a scenario, a claim is delivered in informal, natural language. Carroll provides a simple yet structured form, depicted below, in which a claim states the goals of a particular design feature in addition to the negative effects that might occur as a result of incorporating that feature into the system design [12]:

> *IN <situation>, <design feature> CAUSES <desirable psychological consequence(s)>,*
> *BUT MAY ALSO CAUSE <undesirable psychological consequence(s)>*

In this way, a particular design feature, in a given situation, is depicted as the cause of various psychological effects on the user of the system. Some of those effects represent possible positive outcomes, or opportunities, while others

represent possible negative outcomes, or risks. For the remainder of this thesis, “desirable psychological consequences” will be referred to as “upsides” and signified by a plus (+) sign. “Undesirable psychological consequences” will be referred to as “downsides” or “risks” and signified by a minus (-) sign.

Consider the example claim in Figure 2.1, which states the tradeoffs of relaying information through changing text. This claim might be derived from a scenario about using animated, tickering text in a web interface that describes a user’s ability to easily monitor dynamic information without the need to reload web pages and manually scroll through the text.

This claim states two distinct upsides, while reminding the user to consider the possibility of two downsides. The first upside states that more information can be displayed in a small space by using a form of animated, or changing, text. The second upside states that changing text provides the user with constant updates so that the user always has access to the most current information. A downside of changing text, however, is that only a small amount of text will be visible to the user at any given time, which might make it difficult for the user to understand the message being relayed. Additionally, if the rate of change in text is too fast, the user will find it difficult to read and comprehend the text being displayed.

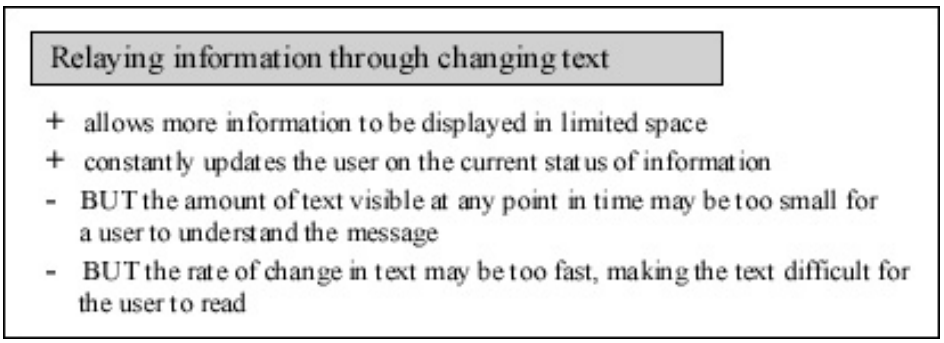

Figure 2.1: An example claim about animated text

A natural language description of the design feature and a list of upside and downside tradeoffs make up the minimal structure of a claim. Additionally, each upside and downside of a claim should be supported, whenever possible, by one or more sources of design rationale. This rationale might arise from an external research publication containing either HCI theory or the results of an evaluation conducted on an existing interface, or design artifact. Rationale might also be generated from the results of an evaluation internal to the current design project. A claim might also provide information about its author, the scenario from which it was derived, the artifact from which it originated, and pointers to any related claims [56].

The validity of a claim must be considered when determining where to focus design and evaluation efforts. When a claim is initially extracted from a particular usage scenario, it becomes a testable hypothesis about the design of a particular interface. In this way, claims are grounded in and validated by their supporting design rationale and through the evolution and evaluation of an associated artifact. Identified claims fail the validation process if their upsides and downsides are not supported by evaluation and use of an associated system. These claims are rejected while substantiated claims become valid, or attested claims [56].

2.1.2.3 Claims-Centric Design

Claims encourage designers to debate design tradeoffs as opposed to accepting a single design principle [54]. The designer's goal is to resolve or mitigate the downsides of each claim whenever possible, while maintaining or strengthening the upsides [14]. In this way, upsides and downsides represent design opportunities and design risks, respectively. However, given the complex nature of design work, designers cannot expect to resolve all possible downsides identified by their claim set. Some downsides will have a highly negative impact on the user's goals while other downsides have a minimal effect.

Although the individual downsides of a claim represent design risks, the risk model developed in this thesis is centered on the claim as a whole. A claim, containing the description of a single design feature and its associated design tradeoffs, represents the fundamental unit of reusable design knowledge. An individual downside risk without its associated claim provides the designer with little context from which to assess the risk and develop an appropriate mitigation strategy. Consequently, claims pay a vital role in the development of the risk model.

2.1.3 Scenario-Based Design Methodology

The concepts discussed above, specifically task analysis, scenario creation, and claims analysis, are combined to produce the SBD process. A designer begins the process by conducting a task analysis. Through field studies, the designer observes a problem space from the perspective of each system stakeholder to determine which tasks the system should help users to accomplish and to examine current solutions for accomplishing those tasks.

Following task analysis, the designer develops problem scenarios that describe the problem space within the context of the user's goals. Through claims analysis, the key features of the current solution space, along with their upsides and downsides, are highlighted and extracted from each problem scenario. The result of this exercise becomes the root concept for the system, which lays out the high-level vision and overall goals for the interface being designed. The designer must keep the root concept in mind throughout the remainder of the design process, namely, throughout the phases of activity, information, and interaction design.

The goal of activity design is to describe high-level system functionality. After describing the user's problems and current solutions in the problem scenarios and claims, the designer develops a set of activity scenarios that describe new solutions for the user's problems. These solutions must maintain the positive aspects of the user's current solutions while mitigating the negative aspects whenever possible. At this stage of the design process, the designer is not concerned with how the user will accomplish specific tasks, but only with determining exactly which tasks the user will be able to accomplish through use of the system [46].

Information design addresses the stages within Norman's Gulf of Evaluation, namely, perception, interpretation, and making sense. The goal of information design is to arrange objects and actions in a way that allows users to perceive and understand them. At this stage, the application's interface displays, menus, dialog boxes, and other sensory information are designed. Information scenarios should elaborate upon these design choices and describe how they allow the user to understand the current state of the system and the functionality that is available to them [46].

The third design phase, interaction design, addresses the stages in Norman's Gulf of Execution, namely, forming system goals, planning action, and execution. The goal of interaction design is to determine how the user will interact with the information designed previously in order to complete a task. Interaction scenarios should specify the mechanisms available to the user for accessing and manipulating the information available in the system [46].

After completing activity, information, and interaction design, the designer typically creates a prototype of the system, which can then be evaluated by end-users. In this way, SBD provides a series of steps that guide the designer through a systematic process to produce a design artifact. However, design is an iterative process. What happens when the user evaluation uncovers a critical flaw in the system design? Should the designer scrap the entire set of scenarios and claims that have been developed and begin the design process again based only on the root concept of the system?

The strategy for system re-design is necessarily dictated by the nature of the problem uncovered. For example, a user's dislike for the placement of a graphical icon will require far less rework than a user's desire for the system to support an additional task. Nevertheless, SBD provides the designer with little guidance for determining how the results of a user evaluation correspond to the particular scenario or claim in which the problem lies or how to address the problem in the next design iteration. To fully support the use of an iterative design process, SBD must continue to guide designers through a systematic process after the creation and evaluation of an initial claim set. To articulate the overall process, as well as the benefits, of iterative design, the following section introduces Carroll's task-artifact cycle.

2.1.4 Task-Artifact Cycle

Carroll's task-artifact cycle, which is depicted in Figure 2.2, describes the iterative evolution of HCI artifacts. When a new system is developed to facilitate achievement of a particular goal, designers first generate user requirements through task analysis and then design a system to support the user in accomplishing the necessary tasks. This design artifact will, in turn, present new challenges to the user, which results in the user's desire to accomplish new tasks [14].

The same fundamental cycle holds true for iterative design. The designer creates an initial solution to the user's problem; the user evaluates the solution and provides feedback. Based on the user's feedback, the designer reassesses the original solution, improves upon the original design, and again asks the user to evaluate the result. This process continues until both user and designer are satisfied with the system. Throughout the design process, the user might determine new tasks that he (or she) wishes to accomplish, reassess the importance of existing tasks, or discover downsides to particular design features that are or are not be acceptable. The designer must take the results of the user's evaluations and determine which aspects of the design must be altered so that the most critical upsides are maintained and the most critical downsides are mitigated.

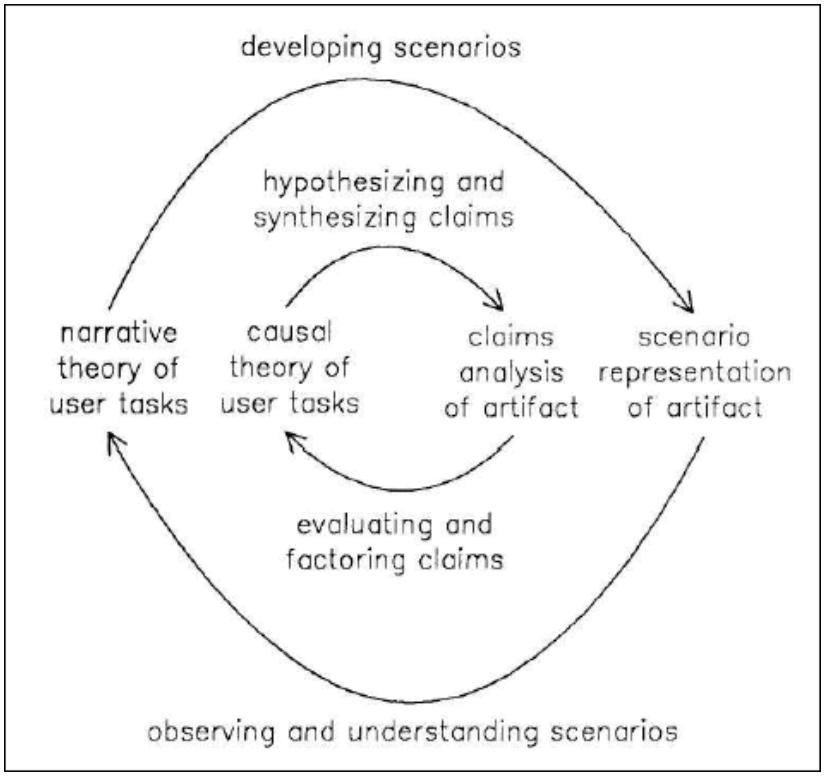

Figure 2.2: Carroll's task-artifact cycle, from [14] (c) 2005 ACM, Inc. Used by permission.

2.1.5 Claim Relationships

As system designs evolve through multiple iterations, claims will be added and removed from the design set and updated to aid in the progression of the design. Additionally, a claim will be updated and improved as it is reused in the design of different systems. In this way, claims evolve through the progression of a single system design and through the reuse of knowledge in the design of multiple systems.

Through the evolution of a single claim, and through the evolution of a set of claims that, together, make up a system design, claims interact and form relationships with one another. Wahid [57] recognizes six distinct types of claim relationships, which are outlined briefly in Table 2.1.

The first two types of relationships, postulating/predicating and executing/evaluating, refer to relationships between claims in different phases of the SBD process. For example, Claim A, which describes the problem domain, has a postulating relationship with Claim B, which suggests a possible solution. Claim C, which relates to the information design of an interface, has an evaluating relationship with Claim D, which relates to the corresponding interaction design. The remaining four relationship types –generalizing/specifying, fusing/diffusing, translating, and mitigating relationships, provide insight into the evolution of a claim across multiple design iterations or through reuse from system to system.

Of particular interest within the context of this thesis is the idea of a mitigating relationship, in which a new claim is created to manage the limitations of an existing claim [57]. Claim G explicitly states its downsides, or design risks, which the designer hopes to resolve. Claim H, which mitigates a downside of Claim G, has a mitigating

relationship with Claim G. In this way, the addition of Claim H to the design set serves to improve the overall system design by mitigating a downside of Claim G.

Sutcliffe and Carroll also recognize the potential for relationships between claims. They describe a claim factoring process, equivalent to Wahid's generalizing/specifying relationship, by which original design claims are abstracted to form more general, reusable claims [56]. The contribution of claims to the reuse of design knowledge is discussed in more detail in the next section.

Relationship	Description
Postulating/Predicating	Postulation and predication relationships exist between claims about a problem situation and claims about a design solution. Postulation relationships lead designers from a problem claim to all possible design solutions; however, in searching a knowledge repository, a designer will not necessarily find the problem claim first. Thus, predication relationships lead the designer from a design claim back to the associated problem claim.
Executing/Evaluating	Execution and evaluation relationships occur between claims used in the information design phase and claims used in the corresponding interaction design phase. An execution relationship leads to a claim related to Norman's Gulf of Execution while an evaluation relationship leads to a claim related to the Gulf of Evaluation.
Generalizing/Specifying	A generalization relationship leads from a claim about one specific concept to a claim about a much broader idea. In contrast, a specifying relationship leads from a broad claim back to a more specific claim.
Fusing/Diffusing	A fusing relationship integrated two claims about different concepts into one high-level, combination claim. A diffusing relationship separates one high-level concept into two distinct claims.
Translating	A translation relationship links two similar claims that belong to different problem domains.
Mitigating	A mitigation relationship exists between a claim with a given downside and any other claim that provides a possible solution to that downside.

Table 2.1: Wahid's six claim relationship types, adapted from [52]

2.2 Design Knowledge Reuse

Design is a creative process; however, few designs are entirely new creations. Instead, designs almost always build upon a foundation of previous knowledge. The concept of design knowledge reuse, itself, developed from the success of reuse techniques in software engineering, as discussed in the next section. The remainder of this section outlines Sutcliffe's Domain Theory, which proposes claims as the fundamental packet of reusable design knowledge, discusses design within the domain of notification systems as a means of furthering knowledge reuse, and describes several tool technologies developed to support SBD and reuse.

2.2.1 Software Reuse

To gain acceptance into current software engineering practices, the science of design must facilitate reuse. However, the concept of reuse is indubitably broad. Within the software domain, reuse has seen considerable success in the form of reusable code modules and object-oriented pattern libraries. However, it is generally accepted

that system developers can reduce development time and cut costs on a larger scale by incorporating reuse at an earlier stage of the development process [21, 54]. Consequently, new problems do not have to be solved from scratch. In the context of a design project, knowledge can be related to the product being designed or to the process by which the design team accomplishes its goals. Archiving and reusing knowledge about a design product and a design process can help to ensure that effective ideas are remembered and that mistakes are made only once. Unfortunately, reuse of user interface components, such as sliders, buttons, and menu bars, is superficial at best. The design knowledge underlying these user interface elements are rarely viewed from the perspective of potential reuse [54].

Numerous software reuse techniques exist; however, each technique can be described in terms of four key dimensions, namely abstraction, selection, specialization, and integration [33]. Abstraction, the key characteristic of any reuse paradigm, involves generalizing a reuse artifact for classification and retrieval. Selection characterizes the mechanism by which artifacts are located, compared, and chosen for reuse. After an abstract, or general, artifact is selected for reuse, it must be specialized, or refined, for use within a new domain. Finally, a set of selected and specialized artifacts must be combined, or integrated, to create one new software system. The effectiveness of an abstraction is measured in terms of cognitive distance, which is the amount of intellectual effort required for the software developer to create a new system from a set of reusable parts. An effective abstraction mechanism minimizes cognitive distance, thus reducing the amount of effort required by the developer [33].

2.2.2 Domain Theory

The pursuit of effective use and reuse of HCI design knowledge raises two key questions [54]. First, how can HCI knowledge be transferred effectively from theory to design in a way that is understandable to designers and other stakeholders who are not experts in cognitive science? Second, how can this knowledge be classified for reuse across domains?

Sutcliffe's *Domain Theory* is a reuse technique grounded in psychology and applied to the domain of interface design. Sutcliffe's goal is to capture design knowledge in a reusable form that can be translated from one designer to another. This knowledge should be equivalent to that contained within the memory sparked when one designer realizes that he (or she) has arrived at a problem similar to one he has previously solved [55]. Sutcliffe proposes that design knowledge be structured as claims and stored in a design knowledge repository, or *claims library*, for future retrieval and application within a new domain.

Claims are anchored in a particular context of use, which often provides a much-needed example to make the application of design knowledge more tangible [54]. However, this anchor also makes a claim more domain-specific. A solution to this paradox requires adequate answers to Sutcliffe's original questions. What makes a claim an effective means of knowledge transfer, and how can claims be reused across domains? Although more general frameworks are applicable to a wider array of domains, they also provide less detailed information than that contained within a claim. Claims are associated with a specific design artifact and supported by one or more sources of rationale. Designers can examine a claim's scenario, user tasks, theoretical foundations, or empirical results and

recognize the claims' applicability to a particular design context. Nevertheless, this knowledge base does restrict a claim's potential for reuse across domains.

Since claims are grounded in usage scenarios, the proper level of abstraction is needed to prevent the loss or misinterpretation of the associated design context. To facilitate reuse, claims must be generalized, classified, stored in a knowledge repository, and retrieved when appropriate for use within a new design context. Effective design knowledge reuse relies on a designer's ability to retrieve relevant claims from the repository based on design characteristics. To overcome differences in terminology used by designers, indexers, and library users, Sutcliffe's Domain Theory proposes the categorization of claims based on two task models: generic tasks and generalizing tasks. Generic tasks represent fundamental procedures with single goals. These tasks focus primarily on cognitive goals, such as Planning, Comparing, and Locating, and they cannot be further decomposed without impairing the achievement of their goal. For example, all variants of the generic *Monitoring* task "observe the environment for state changes of interest"[55]. Generalizing tasks, which are a combination of generic tasks, describe tasks that are easily identified in the real world [55]. These tasks represent typical patterns of activity, such as Progress Tracking, Navigation, and Information Retrieval.

The Domain Theory was a driving force in the initial development of a claims library, which is discussed in Section 2.4. Generic and generalizing tasks were used as a classification mechanism for claims stored in the library. This structure complements the dual-task nature of notification systems, which are described in the next section.

2.2.3 Notification Systems

Design knowledge reuse is a broad problem. To manage the complexity, the work contained within this thesis focuses on reuse within a particular domain –the design of *notification systems*. Notification systems grant users access to additional information from sources outside the scope of their current, primary task.

Notification systems span multiple domains and platforms, from mobile computing devices and in-vehicle information systems to tangible interfaces and large-scale public information displays. Consequently, they provide an abundance of researchable interfaces within a focused application domain. Examples of notifications systems include status monitors, stock tickers, and email alert programs. The key characteristic of a notification system is the secondary nature of the notification task. Information is displayed in the users' periphery without unwanted interruptions to their primary task [37]. The dual-task nature of this class of systems supports knowledge classification through Sutcliffe's generic and generalized tasks, making notification systems a useful domain in the advancement of knowledge reuse. Additionally, classification is supported by critical parameters, as discussed in the next section.

2.2.3.1 Critical Parameters

To allow systems to evolve and improve, effective modeling and evaluation techniques are needed. Newman [38] argues that the development of such techniques first requires the definition and acceptance of *critical parameters*, which provide a measurement of a design's ability to achieve its intended goal. Critical parameters are defined for an individual domain, or class of systems; however, these parameters look beyond the details of any

specific application to focus on the broader purpose of the technology. In this way, critical parameters, when selected appropriately, can serve as benchmarks within their domain.

McCrickard defines three critical parameters for notification systems: interruption, reaction, and comprehension [37]. Depending on the context and usage of a particular system, users will desire the effects of these parameters to various degrees.

The *interruption* parameter measures the degree to which a user's primary task is disrupted by an incoming notification. A system must provide the appropriate level of interruption to alert the user without causing unwanted distractions. In some situations, the user might wish to be interrupted, perhaps due to the arrival of time-sensitive information. In other situations, such as in the use of an in-vehicle information system, unwanted interruption can have a disastrous effect.

The *reaction* parameter measures the degree to which users should respond to a notification. A system must provide enough information for the user to understand the notification and react accordingly, either by taking action or by dismissing the notification. In this case, a timely and precise response to the notification is more important than attracting the user's full attention. For example, an in-vehicle navigation system can notify the user when to make a turn without interrupting the user's primary task of driving the vehicle.

The *comprehension* parameter measures the degree to which the user understands the information provided by a notification system. A system might facilitate long-term comprehension of the notification content, such as noticing trends or linking previous knowledge to new information. For example, after months of monitoring stocks on a secondary display, a user might possess a solid understanding of recent trends and begin to predict future notifications.

2.2.3.2 IRC Framework

For a given notification system, the prevalence of each critical parameter is assigned a numeric value from 0.0 to 1.0, where a value of 0.0 represents the complete absence of the characteristic and a value of 1.0 represents a strong presence of the characteristic. The collective triple of all three parameters is known as the IRC rating for the system. Figure 2.3 illustrates the three-dimensional space constructed to describe the range of notification systems applications. Each corner of this IRC cube corresponds to a particular type of notification system, based on their varying levels of interruption, reaction, and comprehension.

For example, an ambient display (IRC rating: 001) and an alarm system (IRC rating: 110) reside at opposing corners of the IRC cube. A user of an ambient weather monitor wishes to maintain awareness of evolving weather conditions; however, interruption and immediate reaction to changes in the weather are not critical. In contrast, the purpose of an alarm is to interrupt current activity and redirect the user's attention to a new task [37].

An IRC rating can also be assigned to individual claims. This rating characterizes the degree of interruption, reaction, and comprehension supported by the claim's design feature. A comparison of the target IRC rating for a system design and the IRC rating assigned to a claim in the design set provides an indication of how well that claim is supporting the overall design goals. The difference between the two IRC ratings is one factor that should be considered in determining which claims should be the focus of redesign efforts.

The domain of notification systems design combines the simplicity of a single, focused domain needed for the advancement of design knowledge reuse with the complexity of a diverse class of applications needed to challenge the SBD process. The next section describes a suite of interactive tools developed to support the SBD process, the reuse of design knowledge, and the design of notification systems.

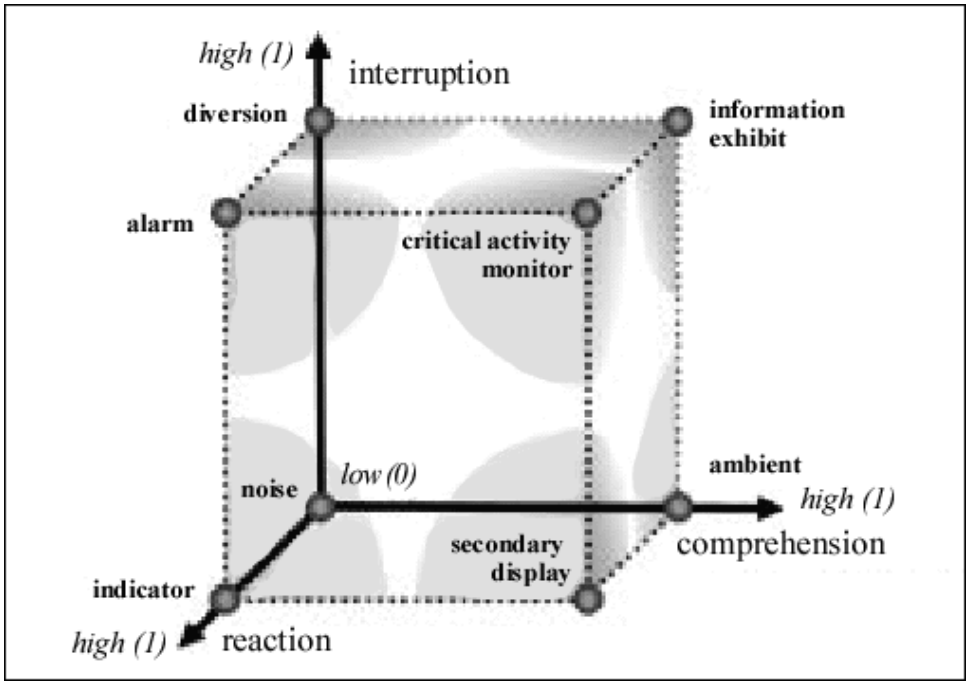

Figure 2.3: A three-dimensional representation of the IRC framework, from [36] (c) 2005 ACM, Inc. Used by permission.

2.2.4 Supportive Tool Technologies

The success of methods for SBD and claims analysis relies on the development of adequate reuse repositories and effective tool support. Tools are needed to evaluate the practicability of these methods and then to facilitate learning and promote practical acceptance in academia and industry. The ongoing development of such tools is discussed in the following sections.

2.2.4.1 Developing a Claims Library

To facilitate *design by reuse*, someone must first *design for reuse*. A collection of reusable claims and a classification mechanism that allows designers to locate and understand the knowledge contained within those claims are crucial. Payne et al [44] designed and implemented a claims library, based on Carroll's SBD process and Sutcliffe's Domain Theory, to support the generation, generalization, and storage of valid claims as well as the development of an appropriate retrieval mechanism. This claims library provides meaningful knowledge about the design of notification systems and encourages designers to search for and consider claims about systems similar to an application they are currently designing. Consequently, the library must store claims at an appropriate level of

abstraction to facilitate retrieval. Although claims made about one system might not have the same effects when applied to another system, acquiring reusable knowledge from the claims library allows designers to identify design alternatives and assess the advantages and disadvantages of each one.

Key claim attributes stored within the repository include the design feature description, upside and downside tradeoffs, sources of design rationale, an associated scenario, and links to related claims. An associated artifact serves as a design abstraction [44]. The claims classification framework supports reuse and aids designers in assessing the relevance of claims for application within a new project [56]. Each claim in the library is situated in an abstract, four-dimensional space, or "footprint," decided by its primary task, notification tasks, IRC ratings, and design abstraction. The primary task describes a broad user goal in the form of a generalized task, while the notification tasks define user goals as low-level, generic tasks. The design abstraction refers to an existing design artifact and consists of a set of keywords generalizing an interaction between system and user. The IRC ratings, presented on a scale from 0 to 1, represent the impact of the artifact on the user in terms of interruption, reaction, and comprehension [44].

Although the claims library could prove useful in isolation, its true utility will result from integration with a collaborative design environment. Such a system would facilitate reuse while supporting design by scenario. The ongoing development of such a system, termed LINK-UP, is discussed in detail in the next section.

2.2.4.2 Leveraging Reuse with LINK-UP

LINK-UP [18] is an integrated design environment that guides designers through a scenario-based usability engineering process. The web-based system is linked to the claims library introduced in the previous section, providing access to a growing compilation of reusable design knowledge. Just as the claims library focuses on design knowledge relating to the design of notification systems, LINK-UP leverages the IRC framework and Norman's concept of a system image [39] to support designers in creating effective interfaces for divided-attention situations. Designers use LINK-UP throughout the design process, from requirements engineering to usability testing. The system currently consists of two key modules for requirements analysis and creation of a system image.

Specifically, LINK-UP helps designers to determine their target, or design model, IRC values based on an assessment of end-user tasks and system requirements. The system then aids designers in locating relevant claims for each usage scenario and stage of action. Once a system prototype, or design artifact, is created, LINK-UP helps designers to evaluate the design and acquire the actual, or user model, IRC values for the system. Following an evaluation, designers can continue to use LINK-UP in redesigning their interface. However, the structured design process that LINK-UP provides throughout the first design iteration diminishes significantly as designers approach redesign. An overall goal of this thesis is to collect key pieces of information from each LINK-UP module throughout the first design iteration and use that information to provide more structure to the redesign process.

2.3 Project Management in Software Projects

The LINK-UP system can be used by individual designers. However, given the inherent complexity of the design process and the increasing size of interactive systems, teamwork is, in many situations, becoming the

preferred method of design. Thus, LINK-UP, as a collaborative design environment, must support design teams in managing their projects. Key management tasks, including maintaining team coordination, monitoring progress, and managing risk, are crucial to the success of team-based projects. Unfortunately, many software projects fail because they are not managed properly [31].

2.3.1 Modern Team Dynamics

As the size and complexity of software-intensive systems continues to increase, it has become difficult for one individual to achieve a full understanding of all aspects of a system design. The knowledge and expertise necessary for successful design is typically distributed among a group of individuals who must share their knowledge, coordinate their efforts, and resolve conflicting perspectives to solve a given problem. Consequently, individuals rely on effective teamwork, sound management, and adequate tool support in the design of complex, interactive systems. A team is commonly defined as "a collection of individuals who are interdependent in their tasks, who share responsibility for outcomes, who see themselves and who are seen by others as an intact social entity embedded in one or more larger social systems, and who manage their relationship across organizational boundaries" [6]. Successful teams typically have well-defined roles, strong leadership, open communication, and respect for the skills and ideas of everyone on the team. They are committed to their project, establish clear goals, and assess their processes and evaluate their progress early and often [20]. These characteristics apply to both co-located and distributed teams.

Team collaboration refers to the ways in which team members work together in accomplishing a shared goal. Effective collaboration requires team coordination and risk management. Coordination involves establishing team cohesion, maintaining awareness about the activities and perspectives of everyone on the team, and managing dependencies among project-related activities [36]. All members of a well-coordinated team not only share the same knowledge, but also *know* that they share the same knowledge. Consequently, these teams spend less time discussing process-related issues of *how* goals should be accomplished and more time discussing product-related issues of *what* goals should be accomplished [24]. Risk management, which is discussed in detail later in this chapter, involves identifying and prioritizing potential problems and monitoring, mitigating, and controlling those risks throughout the life of a project. To accomplish these goals, the members of a team must maintain a shared understanding of all project-related knowledge, which should include knowledge about the past (i.e., what happened in previous projects), the present (i.e., what is happening in the current project), and the future (i.e., what could go wrong as the project progresses).

Globalization has further complicated the issue of project management by popularizing the use of organizationally and geographically distributed teams. Unlike traditional, co-located teams that share a physical workspace, distributed teams have the added difficulty of collaborating across the boundaries of space and time. Design work, in a traditional sense, often takes place with a small team of designers working together within a shared space. However, as systems and organizations continue to grow, design teams will also be forced to increase in size and distribution to meet the needs and constraints of a project. Regardless of whether teams are distributed by time zones or merely by conflicting schedules, adequate management support is crucial. This support is twofold; it

requires a solid management paradigm, defined and adopted to meet the needs of a specific project and design team, and effective collaborative tools that support the central facets of that paradigm.

2.3.2 Existing Project Management Tools

Techniques for managing fully or partially distributed teams have not been fully explored; however, it is generally accepted that these virtual teams cannot be managed using traditional paradigms. Although managers of distributed teams face the same challenges as managers of traditional, co-located teams, including establishing and maintaining team cohesion, sustaining momentum, and monitoring quality and progress throughout the life of a project, these challenges are exaggerated when team members are unable to meet face-to-face [43]. Consequently, virtual project managers must develop new techniques to effectively manage their teams.

Many existing tools attempt to facilitate team collaboration; however, most of these systems do not adequately support critical aspects of project management. Four key management concerns, namely system accessibility, reduced project overhead, activity awareness, and risk management, are outlined in the following sections.

A handful of existing systems are also introduced in the following sections as examples of both adequate and inadequate support for these management concerns. Some of these tools are tailored for use within the software development domain; others are targeted toward other domains or generalized for use by a variety of work groups. Some of these systems are targeted toward student software engineering groups, while others are designed for use by distributed teams in research and industry. The systems discussed in this chapter are not intended to encompass the full array of available tools; however, they do represent a progression of project management tool support for collaborative teams and illustrate the need for improvement in management tool support within the software domain.

2.3.2.1 Maintaining System Accessibility

The first key management concern is the convenience and accessibility of a collaborative system. A system that is to be adopted and used regularly by all members of a project team must be made easily available to those team members, regardless of their physical location. The convenience and accessibility of Internet technology constitute obvious advantages, and web-based applications are being deployed increasingly often to support the widespread and highly mobile nature of computer users. This trend continues within the domain of collaborative systems and has been accelerated by the increased distribution of project teams.

Web-based tools are available from any device capable of running a web browser. In this way, the flexible and ubiquitous nature of the web facilitates the development and use of web-based tools in support of mobile and globally distributed teams. Team members working from home, from customer sites, or from any other web-accessible location, can connect with teammates without the need for elaborate software or network access. With web-based tools, team members can access and update project information from almost anywhere in the world.

Making project information available through an online system also aids in team organization and ensures that the same information is available to everyone involved with the project. Additionally, it reduces the time required for team members to gain feedback from one another, from project managers, and from other project stakeholders. Project stakeholders can receive updates almost instantaneously, edit electronic documents, and send feedback

quickly and easily. Based on the benefits outlined in this section, effective project management tools must take advantage of the convenient and widespread accessibility gained through the use of web-based applications.

2.3.2.2 Minimizing Project Overhead

Coordinating tasks among teammates and monitoring progress require significant time and effort. However, collaborative systems only support these tasks if everyone on the team uses the system consistently. Unfortunately, the effort required in using many collaborative systems is not always rewarded. Grudin observed [26] that many collaborative tools, such as meeting scheduler systems, fail because the individual who performs the majority of the labor in using the system is rarely the same individual who receives the majority of the benefit in using the system. To schedule a meeting, an individual selects the meeting participants, and the system automatically compares the participants' schedules to suggest a mutually available time slot. Unfortunately, this scheduling technique is only effective if all of the meeting participants maintain an accurate personal calendar within the system. Consequently, the benefit realized by one individual depends on regular use of the system by a number of other individuals, who might never realize a comparable benefit.

A similar situation occurs when project managers attempt to monitor progress by requiring individual team members to maintain a list of their assigned project tasks. The outcome of this situation is illustrated by the Student Online Project Planning and Tracking System (SOPPTS), a web-based, task-oriented project management system for use by undergraduate software engineering teams [60]. At the start of a project, teams produce a list of project tasks and assign subsets of those tasks to each team member. Team members are then responsible for updating the system as progress is made on each task. Consequently, all team members and the project manager can see which tasks have been completed, whether each task was completed on time, and if certain tasks, or team members, have fallen behind schedule.

In accordance with Grudin's paradox, SOPPTS proves beneficial for teams only when the system is used regularly and by all members of the team. One major shortcoming of the system is that it requires a high and consistent level of input from each team member. Students must input their initial set of tasks and then update their progress on each task regularly in order for the system to provide an accurate view of team progress. The amount of overhead that use of SOPPTS adds to a project in terms of consistently updating progress on individually assigned tasks can distract team members from other project tasks and, in some cases, actually hinder progress. Consequently, use of the system typically diminishes as a project progresses.

To fully benefit from a project management tool, project team members must be able to manage their work without being unduly distracted from other project tasks. To this end, an underlying goal of this thesis is to inject management techniques, to the greatest extent possible, directly into the design process. As a result, project management will begin to occur as a natural byproduct of following a design process, as opposed to adding unwanted overhead to a project.

2.3.2.3 Providing Activity Awareness

To collaborate effectively, project teams must communicate and remain aware of each member's actions. Teams using a collaborative environment can benefit from virtual cues about team activities (i.e., what are my teammates doing?), availability (i.e., who is currently available to be contacted?), processes (i.e., what are my task requirements and deadlines, and how are they affected by the requirements and deadlines of my teammates?), and perspectives (i.e., what are my teammates thinking and why?). *Activity awareness* [16] is defined as awareness of other people's plans and understandings, with respect to a long-term, collaborative undertaking with substantial goals. When team members are aware of the current status and actions of the individuals using a collaborative system, they possess a better understanding of how pieces of a project must fit together to achieve the project goal [30]. Existing collaborative systems support activity awareness to varying degrees through the use of notification systems, which can provide an abundance of awareness data to the members of a distributed team without distracting them from their primary tasks. However, most collaborative systems do not take full advantage of these benefits.

TeamSCOPE [53], for example, targets distributed project teams with the goal of improving awareness among team members in a collaborative work environment. The system provides teams with a shared file repository, dedicated message boards for each shared file, and a detailed activity history. These features improve collaboration by allowing users to monitor their teammates' recent, relevant activities, i.e. those activities that relate to their own current tasks, without being inundated with information about all recent activities. However, the organization of the activity history into a list of recent events hinders a team's ability to see the project as a whole and allows team members to get lost in the details of current tasks. At login, users are presented with list of the most recent activities. The list is ordered chronologically; however, structuring the information along a project timeline could greatly increase the amount of insight gained through the visualization.

TeamSpace [25], which supports the synchronization and documentation of team meetings, organizes information presented during a meeting into a timeline. Key events, such as a team member arriving, leaving, presenting important information, or making a decision, are recorded using descriptive icons. These icons can then be filtered by type or selected to access further details. Team meetings are only one type of event that can then be included on a full project timeline along with deadlines and other project milestones. Structuring project-related knowledge according to the common dimension of time exploits our ability to organize past experiences into a sequence of episodes and aids team members in maintaining an overall view of the project.

As team members work together in the design of system interfaces, they must maintain awareness of one another's current activities, as well as activities that have taken place previously in the overall evolution of the design. The use of project timeline visualization, similar to the technique employed in TeamSpace, can be integrated into LINK-UP to aid designers in maintaining an overall view of their project, in assigning tasks to individual team members, and in monitoring progress throughout the life of a project. This timeline can also aid teams in managing risk, which is the fourth management concern discussed in this chapter. The lack of support for risk management in existing collaborative tools is discussed in the next section.

2.3.2.4 Supporting Risk Management

All projects have risks; however, none of the tools discussed in this chapter explicitly incorporate risk management, either exclusively or in conjunction with other project management capabilities. Although a number of risk management techniques and a small subset of supportive tools do exist, the majority of collaborative tools overlook the need to manage risk. This lack of support for risk management in collaborative systems, and specifically in software management tools, ultimately motivated and focused the goals of this thesis.

Project risks must be identified early in the project life cycle and monitored continuously throughout the course of the project. Existing tools, such as SoftRisk [32] and @Risk [42], support risk identification, analysis, and monitoring; however, both systems have limitations. SoftRisk identifies risk through a set of generic checklists and questionnaires, which cannot be customized to address the needs of a specific project. In contrast, @Risk calculates risk according to one of many probability distributions. However, the system requires users to develop their own set of risk parameters for a given project. The correct balance between all-purpose and customized tools is needed to address these issues. Following an introduction to managing risk in the next section, these and other risk management tools and techniques will be described in more detail.

A review of existing collaborative tools, presented in abbreviated form in the preceding sections of this chapter, lead to the construction of four guidelines for the development of effective project management tools. One of these guidelines specifically addresses the importance of supporting risk management. The successful implementation of a risk-driven model to facilitate management in software design projects would not only expand upon this risk-centric guideline, but also facilitate achievement of the three remaining guidelines, which discuss the need to support activity awareness, visualization of project evolution, and reuse of project-related knowledge. These guidelines will be outlined in more detail in the next chapter. The remainder of this chapter focuses on the foundations of risk management and current standards within the software domain.

2.3.3 Managing Project Risk

Risk management is a critical, yet often overlooked, aspect of software projects, and, more specifically, of HCI design projects. Managing risk, like most other management tasks, typically adds unwanted overhead to a project. Consequently, project risk is often managed in an ad hoc fashion, or, in some cases, not managed at all.

All projects have risks. A risky action or event involves an element of uncertainty or chance, an associated loss, and a choice to be made about how to handle the risk [17]. Some risks are more probable, influential, or costly than others. In fact, each member of a group of project stakeholders might possess a different opinion concerning the loss associated with a certain risk, the level of uncertainty involved, or the choice that should be made. Risk management involves identifying and prioritizing potential problems and monitoring, mitigating, and controlling those risks throughout the life of a project.

Risk management is a broad and complex issue. The next section serves to acknowledge this breadth and then to narrow the scope of the risk domain for the purposes of this thesis. Section 2.5.3.2 discusses existing risk management techniques, along with the strengths and weaknesses of supporting tool technologies, as they apply to the software domain. Following this presentation of the foundations of risk management, Section 2.5.3.3 outlines the

direction of this thesis toward the application of a risk-driven, claims-centric management model to guide the design of interactive systems.

2.3.3.1 Foundations of Risk Analysis

Project risk is a common consideration in many areas of business and industry. Insurance specialists, for example, assess accident statistics. Financial organizations assess credit and investment risk. Physicians assess the risks associated with various treatments for a patient with a particular injury or disease. In all cases, risk management involves determining what could happen in the future, analyzing associated risks and uncertainties, defining alternative courses of action, and making a decision based on all available information [5].

Uncertainty is typically defined as "a lack of ability to accurately predict the outcome of a performance measure" [5]. In some risk domains, such as economics, a distinction is made between uncertainty and risk. In other areas, such as project management, the two terms are synonymous. This combined definition of risk, or uncertainty, is often restricted to encompass only possible negative outcomes. In this case, possible positive outcomes are termed 'opportunities.' In recent years, the term 'risk' has acquired the connotation of a dangerous or dreaded event, and this work, like that of MacCrimmon and Wehrung [34], and others, focuses primarily on the negative aspects of risk. For the purposes of this thesis, *risk* will be defined according to Charette's definition [17], which was originally stated in Rowe's book, *An Anatomy of Risk* [47].

> *Risk is the potential for realization of unwanted, negative consequences of an event.*

Uncertainty is measured and quantified as a probability, based on the knowledge available and the viewpoint of the evaluator [5]. Available knowledge includes historical data, background information, system performance characteristics, and other project assumptions. Probability is only determined objectively in situations, such as gambling, when all possible outcomes can be assigned the same probability. In most cases, probability is determined subjectively, either as an estimate based on relevant empirical data or as a measure of the degree of belief that the predicted outcome is, in fact, the true outcome. Uncertainty is often assessed by calculating appropriate probability distributions and making predictions based on those estimates.

The overall goals of risk management are to identify all possible project risks, prioritize those risks, mitigate the most critical risks, and monitor all risks throughout the course of a project. The work contained within this thesis focuses primarily on risk prioritization and the method of calculating individual risk exposure values. Risk exposure, or risk weight, is typically calculated according to the probability that the risk event will occur and the impact that the event will have on the project if it does occur. Thus, for a simple risk model,

$$\text{Exposure (risk event)} = \text{Probability (risk event)} * \text{Impact (risk event)}.$$

In business, economics, medicine, and some areas of industry, risk management is supported by a solid foundation of knowledge and techniques. However, the software domain, a field that remains in relative infancy

compared to other fields, lacks this foundation upon which to build. Although software professionals attempt to derive effective techniques from proven methods and models in the business world, risk management remains a groundbreaking practice throughout much of the software domain.

To further develop risk methodology in software project management, Wallace et al. [58] define six dimensions of software project risk, which are outlined in Table 2.2. Most of the factors considered in this model correspond to process-related issues, such as lack of user involvement, poor project scheduling or cost estimation, and team communication concerns. Issues surrounding system requirements, including incorrect, ambiguous, or frequently changing requirements, begin to address the risks related to the product being developed; however, none of the six software risk dimensions specifically focus on product-related risks.

Category	Description
Organizational Environment Risk	Risk or uncertainty surrounding the organizational environment in which the software project takes place, including factors such as organizational politics, stability of the organizational environment, and organizational support for the project
User Risk	Risk related to the user or other project stakeholders, including lack of user involvement and unfavorable user attitude toward the project
Requirements Risk	Uncertainty surrounding system requirements, including incorrect, unclear, inadequate, ambiguous, unusable, or frequently changing requirements
Project Complexity Risk	Inherent complexity difficulty of the project being undertaken, indicated by factors such as the degree to which new technology is being used, the complexity of the processes being automated, and the number of required links to existing systems and external entities
Planning and Control Risk	Increased project risk due to poor planning and control, which often lead to unrealistic schedules and budgets, lack of visible milestones
Team Risk	Issues associated with project team members, such as team member turnover, insufficient knowledge among team members, and cooperation, motivation, and communication issues

Table 2.2: Wallace's six dimensions of software risk, adapted from [53]

The work contained within this thesis seeks to expand this model to include a seventh dimension, highlighting product-related design risks. This work will capitalize on the framework developed by Wallace et al. to aid in defining a product-related risk and in comparing the structure of process and product-related knowledge. Although this thesis focuses on managing product-related risks in HCI design, an underlying goal is to facilitate the integration of process and product-related risks in future work.

2.3.3.2 Risk Methodology in Software Development

Software development teams are plagued by management problems that result in missed deadlines, budget overruns, and canceled projects, and effective management remains an open problem as development teams struggle

to keep pace with changing technology [45]. The overall goals of risk management are to assess and control risk throughout the course of a project. The exact process of effectively managing risk is defined differently by various sources; however, the basic steps remain the same. As outlined in the Software Engineering Institute's (SEI) Risk Management Paradigm [3], which is depicted in Figure 2.4, risks must be identified based on customer goals, assumptions, and project constraints, and analyzed to determine both the magnitude of the potential problem and its likelihood of occurrence. The project team must then plan for risk avoidance and reduction, track changes based on reassessment and corrective action, and, ultimately, control risk through these management steps and through effective team communication.

Figure 2.4: The SEI Risk Management Paradigm, from [3] (c) SEI 2005, used by permission

Given the increasing complexity of software-intensive systems and the inevitable limitations on time and project resources, teams cannot expect to mitigate or resolve all possible project risks. Risk prioritization allows team members to identify and focus on the most critical risks at any time during the project. The probability-impact equation for calculating risk exposure provides a simple model; however the method of assigning quantitative values to the probability and impact factors is rarely a simple process. Moreover, researchers often brush this complexity aside, presenting the basic risk equation and waving their hands at the actual calculation of values.

Risk assessment and estimation is the second step in the basic risk management process; thus, without an adequate method of quantifying and calculating risk exposure, project teams will not be able to advance very far through the process to achieve the benefits of managing project risk. Nevertheless, most existing risk models ambiguously define these calculations, leaving developers to define their own, often unstructured, methods of calculating risk exposure values. Other models base judgment on purely subjective measures, a practice that can result in extremely unreliable responses. For example, the interpretation of a major effect compared to a minor effect on a five-point scale might vary significantly among individuals responding to a specific question [35]. In many cases, some form of subjective measurement is necessary. Nonetheless, a key goal in the development of a risk-driven model for HCI design is to clearly define a method for calculating design risk exposure and to incorporate measurable, objective values whenever possible.

A number of techniques have been developed in research and industry to apply risk management models to the software domain. Some techniques mirror those used to manage risk in business and engineering while others attempt to address the distinctive features of software projects. Many techniques adapted from the financial risk sector calculate risk exposure based on probability distributions and simulations [22]. Although probability distributions are useful in estimating risk, they typically require managers to develop their own models based on attributes specific to their projects. Probability distributions and simulations allow managers to make predictions at the start of a project. Unfortunately, these techniques are not effectively integrated into the development process. Consequently, they are often conducted at the start of a project but are not monitored and updated as the project evolves.

Risk management is often viewed as a task for the project manager. However, the knowledge needed to effectively assess project risk is typically dispersed among a group of people in and around the project. Consequently, a team of risk managers can improve the quality of a risk assessment by bringing diverse knowledge and perspectives to the decision-making process and by correcting the biases that result from an individual's specialized role within a project. When an individual assesses project risks, he (or she) makes a judgment based on what he sees and what he knows, or more accurately, "he combines that which he thinks he sees with what he thinks he knows" [27]. Risk assessment is not a simple procedure, and individuals base their assessments on knowledge that they selectively store and retrieve from memory. As a result, individuals are likely to form different judgments, biased by their individual interpretations, selective attention, and selective memory. This bias can be corrected by taking advantages of group performance that is applied effectively to a risk assessment task.

Other risk techniques, such as the Delphi method [1], also take advantage of the collaborative nature of project work. Using this iterative voting process, a group of project stakeholders individually assign probabilities to a given factor. These probabilities are then displayed anonymously to the group, allowing everyone to see the distribution of values. After viewing the results, but without discussing them with one another, the members of the group reassess their individual probabilities, perhaps updating the value based on the results of the previous iteration. This process continues until group consensus is reached or until no change occurs for several rounds.

As risk methodologies continue to expand into the software domain, more tool technologies are being developed to support software teams in managing project risk. SoftRisk [32], for example, is a research tool making strides toward risk automation. SoftRisk is dedicated to managing risk in software development projects and aids software developers in risk identification, prioritization, and monitoring throughout an iterative project lifecycle. Based on responses from a set of checklists and questionnaires, potential risks are identified and assigned a risk exposure value. Risk exposure is determined by both the probability that the risk will become a problem during the life of the project and the impact that the risk will have on the project if it does become a problem. Risks are then prioritized according to their risk exposure values. The tool visually monitors changes in risk priority throughout the life of the project. SoftRisk's time-based visualization of risk evolution is shown in Figure 2.5.

SoftRisk was developed for use with any size or type of software project; thus, the specific benefit for any single domain is limited. Elements of the checklists and questionnaires used in identifying and estimating risks are necessarily general. Additionally, the risk management process is not integrated with other management techniques

or with the software development process in general. Despite these limitations, the underlying concepts that drive SoftRisk are fundamentally important. Applied to a more specific domain and integrated within a collaborative design environment, a risk management tool such as SoftRisk could provide significant benefit in terms of project management.

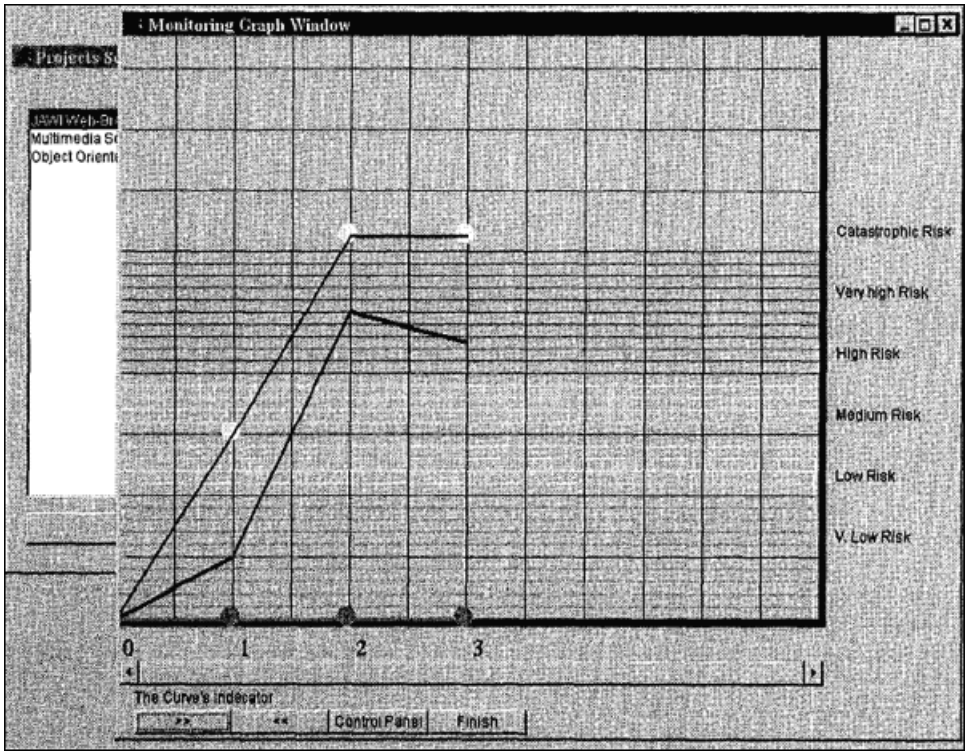

Figure 2.5: The SoftRisk visualization interface, from [32] © 2005 IEEE. Used by permission.

2.3.3.3 Risk Management in Practice

Common process-related risks in industrial project teams, and especially in student project teams, include poor team formation, inflexible deadlines, inappropriate solution paradigms, risk-insensitive progress metrics, tendencies to promise too much too soon, tendencies to focus on a single aspect of the design and lose sight of the project as a whole, and reliance on the thrill of crisis management [10]. These problems typically result from the education and use of cookbook solutions to the design of software intensive systems. They can cause teams to develop good solutions to the wrong problems, leading to the need for large amounts of rework late in the development cycle. Incorporating risk management into the computer science curriculum can aid students in addressing these and other problems. Risk management teaches students to focus on stakeholder goals, evaluate alternative solutions to potential problems, and resolve risks early to avoid late rework. A vital first step toward managing project risk is to make risk visible to all members of a project team. Once team members see and understand their project risks, they must monitor, prioritize, and mitigate those risks early and often.

Boehm advocates the use of top-N risk lists in student software engineering courses. In particular, student project teams are required to identify initial project risks, maintain a prioritized list of their top-N risks, and track the top-N risks, through regular reassessments, throughout the course of the project. By incorporating risk management into a long-term team project, students not only learn risk management concepts through lectures and reading assignments, but also learn risk management practice through integrating project work. By maintaining a top-N risk list, students can track changes in priority over time and focus on mitigation strategies for the most critical project risks at any given time. A sample risk list is presented in Figure 2.6.

Risk Items	Weekly Ranking			Risk Resolution Progress
	Current	Previous	# Weeks	
COTS mismatch	1	5	8	Push for early installation of all COTS packages and test its functionalities
Availability of Rational Clearcase 4.0 for NT and Microsoft Access database for on time delivery	2	10	4	Contacted Rational regional representative and notified him that we are on a time-constraint schedule
Schedule—an independent variable, delivery in 12 weeks	3	2	4	Prioritize requirements and use stage delivery to avoid schedule crunch
Budget—man-hours to be put in by the project team	4	3	4	Use familiar tools, use COTS packages, and add additional team member
One team member will not be available for one week in March, 2000, and another in May 2000	5	6	4	Let other team members help in his area
Poor communication with customers	6	9	3	Schedule weekly meetings. Use teleconferences, emails to facilitate communication
Requirement mismatch	7	6	2	Provide updates to stakeholders and collect inputs during reviews

Figure 2.6: An example of Boehm's top-N risk list, from [10] © 2005 IEEE. Used by permission.

In industry, project managers monitor risk, quality, and progress through the use of software metrics. However, the design phase of a software project is characterized by creative evolution and frequent change. Consequently, metrics applied to software projects during the design phase are far less accurate than those applied later in the life-cycle [49]. In this volatile stage of development, project teams must focus on managing project risk in relation to the overall evolution of the design. Instead of monitoring a dashboard of fluctuating metrics throughout the design phase, project managers might benefit from the ability to project managers could gauge design progress according to set deadlines, monitor design stability, and determine which team members are contributing to which aspects of the design.

If information related to project risks was available to all members of a team, then everyone could monitor, discuss, and implement process problems and improvements. Figure 2.7 presents a typical implementation of the SEI Risk Management Paradigm from [59]. In this cyclic model, new risks are identified, evaluated, and classified for storage within a risk database. Risk prioritization, mitigation strategies, and status information are then stored

with the risk in the database. Stored information is presented to the project team in the form of a prioritized risk spreadsheet, similar to Boehm's top-N risk list discussed in the previous section.

According to the risk database model, electronic classification and storage of risk-related information ensures that the information is accessible and easy to interpret. In this way, a risk database, when implemented effectively, quickly compensates for the cost of initial setup. However, under the SEI model, a risk database maintains data only for a single project. Information about risks identified and mitigated in previous or related projects is not retained for use by subsequent project teams. Given the frequent repetition of common problems in software projects, the reuse of risk-related knowledge could be invaluable. This concept, in relation to reuse of HCI design knowledge, will be a key focus of the next chapter.

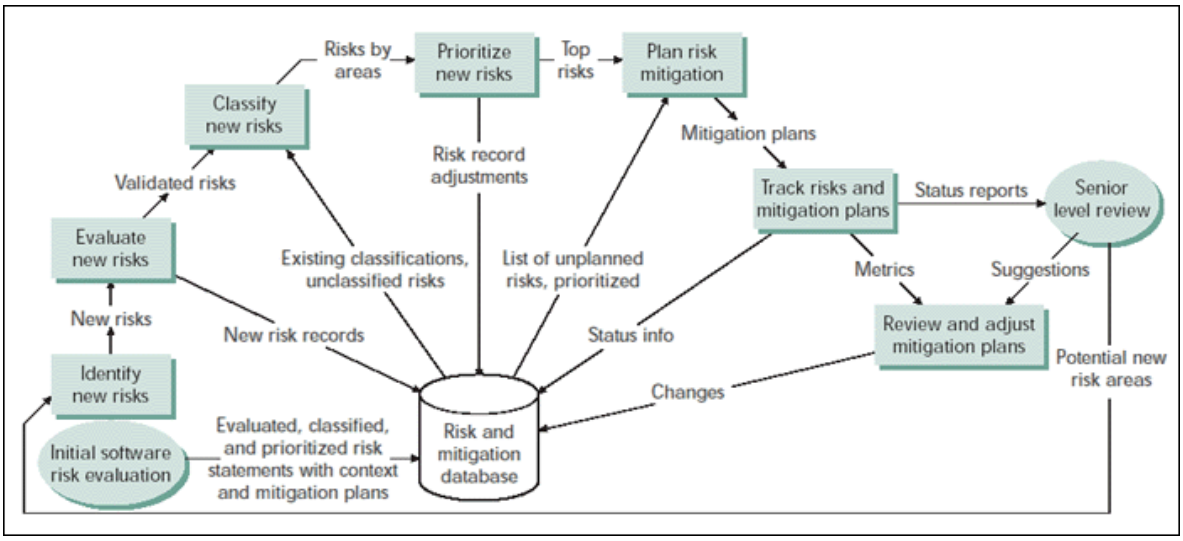

Figure 2.7: A typical implementation of the SEI Risk Management Paradigm, from [54] © 2005 IEEE. Used by permission.

2.3.3.4 Toward Incorporating Risk Management into HCI Design

To manage risk effectively, teams must discuss potential problems, agree on the priority of key risks, assign responsibility for risk mitigation, and monitor progress throughout the project. To accomplish these goals, the members of a team must maintain a shared understanding of all project-related knowledge. Although some traditional, co-located teams can manage risks with written documentation or the use of a simple risk database, distributed teams require adequate methodologies and effective collaborative tools to support communication and to aid in the identification and management of project risks.

To aid effectively in the identification and analysis of project risk, the factors included in a risk-driven model must capture the key dimensions of risk for the domain to which the model is being applied. Additionally, factors should be relevant to the purpose for which the model is being designed, should satisfy common sense, and should have intuitive appeal for the users of the model [35]. To the greatest extent possible, risk management techniques should be integrated with other project management tasks. The management effort, as a whole, should then be

injected directly into the design and development cycle to increase effectiveness and productivity in developing both product and process.

2.4 Summary of Related Work

The work contained within this thesis spans multiple areas of computer science, from knowledge management and software reuse to software engineering and risk analysis to software design and HCI. This chapter is not meant to be a thorough investigation of all of these topics but, rather, an overview of each area, emphasizing the ways in which it contributes to this research.

Scenario-based design and claims analysis are design processes motivated by the quest for a science of design as well as the efficient reuse of design knowledge. These goals lead to the development of LINK-UP and a claims library both to guide the design process and facilitate knowledge reuse. This research effort focuses on the design of notification systems, which support access to information in dual-task situations. Use of LINK-UP by teams of designers, along with the increase in system complexity and team distribution in academia and industry, highlights the need for project management support in collaborative design environments. The need to facilitate management without adding overhead, along with the similarities between risk and the concept of a design claim, encouraged the development of a risk-driven, claims-based management model to be integrated directly into the SBD process. The risk model, which is the main contribution of this thesis, is guided by foundations of risk management that span multiple domains throughout business and industry.

Chapter 3 : Integrating Risk Management into Scenario-Based Design

The investigation of design and management paradigms in the previous chapter identifies the need for improvements both in the process of designing software-intensive systems and in the management of software project, with overall goals of:

- developing a more scientific approach to design
- establishing a higher standard by which software projects are managed

Given the similarities between the traditional concept of risk in software projects and the idea of a claim in SBD, the development of a risk-driven, claims-centric management model has the potential to contribute toward solutions to both of these problems. Identification and prioritization of project-related risks adds structure to the design process by helping designers to develop rationale for why certain project downsides are more critical to address than others. Consequently, the process of determining which design issues to address during the redesign phase of a project becomes more systematic. Furthermore, use of a risk model aids teams in allocating design tasks and monitoring both team progress and the evolution of project-related risks throughout the course of a project. If integrated properly within the SBD process, the risk model with should add minimal overhead to a project.

This chapter discusses the integration of such a model into the SBD process. The next section outlines the overall goals of injecting project management tasks into the design process, based on the shortcomings of existing collaborative tools discussed in Chapter 2. Section 3.1 concludes with the presentation of four guidelines for management tool support. The remaining sections in this chapter elaborate on the first of these guidelines, as it relates to risk management in SBD. Section 3.2 draws a comparison between claims and project risks, elaborating on the differences between product and process-related knowledge. Section 3.3 outlines the overall contributions of incorporating risk management into the SBD process as well as the advantages that SBD provides in managing risk. The section goes on to discuss the use of a risk-driven, claims-centric model to guide the re-design of a system interface. The information available at various stages of the design process is considered in terms of its efficacy in the assessment of design risk. Finally, Section 3.4 summarizes the key contributions of the chapter, reiterating the scope of this thesis and outlining the goals of subsequent chapters.

3.1 Managing Collaborative Design Teams

The work contained within this thesis is motivated not only by the need to improve the management paradigm to meet the evolving needs of software teams, but also to further the science of design. Project management, like

HCI, is a complex discipline in need of a more systematic approach, and effective risk management is a critical step toward achieving both the science of design and the science of software project management.

SBD has made considerable strides toward a science of design with a systematic process of applying theory to design and with a foundation for facilitating knowledge reuse. However, SBD does not provide the same level of guidance for an iterative design process, a significant drawback given the increasing complexity of design projects and the demand for an iterative process.

Risk analysis is traditionally based on a sequence of steps and a strong foundation in mathematics. In other words, it is a scientific process. However, its current application within the software domain, like many other project management tasks, is ad hoc at best. To be effective, risk management must begin early in a project life cycle and continue throughout the course of the project. Unfortunately, risk management tasks are not incorporated into the development process. Consequently, they add significant overhead to a project, and, as projects fall behind schedule, it is these and other management tasks that are often eliminated.

Injecting risk management directly into the design process contributes to both of these issues. Management overhead is minimized without eliminating management tasks. Additionally, the mathematical foundations of risk analysis contribute to the systematic and scientific structure of the design process.

3.1.1 A More Systematic Approach to Design - Prioritizing Design Risks

The goal of risk analysis is to determine which risks are most likely to occur and will have the most impact on the project if they do occur. These high-priority risks are mitigated early and monitored continuously to ensure that their effect is never realized. Other, lower priority risks are also monitored throughout the course of a project to detect changes in their likelihood or impact and to reprioritize accordingly. The result of this re-prioritization is that project teams can remain focused on the most critical issues, even as those issues evolve over time.

The same evolution occurs in many design projects. Designers obtain initial system requirement from the end-user or other project stakeholders, create an initial design to meet the user's needs, and present their solution to the user for feedback. In response to that feedback, the designers conduct a second design iteration to improve their initial solution and, again, present an interface to the user for evaluation. With each design iteration, more information becomes available to the designer, and, consequently, the level of uncertainty, or risk, associated with the project diminishes.

Each risk associated with a design project carries with it a weight, or exposure value, based on the probability that it will occur and the magnitude of its impact if it does occur. The weight of each risk will depend, in large part, on the needs of the project stakeholders. Particular design features and project tasks will be critical to the success of the system, while others will be considered less important. For example, in the design of a progress monitor for use by a team of magazine editors, the choice of colors for individual progress bars displayed on the interface might be of little importance to the project stakeholders; however, designing the system for use on a large-scale, secondary display that is accessible to everyone in the office is critical. The risk of falling a few weeks behind schedule might not be considered a problem as long as the resulting system is delivered according to specification and within the allotted budget.

The weight of each risk changes according to the specifics of each individual project; however, information gathered throughout the design process, from requirements analysis to user evaluation, contributes to the determination of this risk exposure value. As these risks are prioritized and reprioritized during each phase of the project, the designers obtain a clearer picture of the user's needs, how those needs are changing over time, and when those needs and their resulting design are beginning to stabilize. More importantly, at each stage of the design, the project team can remain focused on the most critical design issues that must be addressed during the current design iteration. These high-priority risks must be mitigated and monitored, and the results of the mitigation must be evaluated to determine if the action taken actually improved the system being designed. In this way, integration of risk management into the design process serves to guide the re-design and improvement of a system as well as the evaluation of that system to determine how well it meets the user's needs.

3.1.2 Raising Standards for Project Management - Maintaining a Team Memory

Project constraints, such as system requirements, time, and other project resources, will typically prevent the design team from mitigating every risk associated with a project. Consequently, maintaining a prioritized list of project risks can improve design efficiency and increase the probability of delivering a successful system. When supported by adequate tools and techniques, a risk-centric view of a design project might also help to solve other issues in managing collaborative teams, such as maintaining awareness and simplifying task assignment and visibility. When dealing with the design of large, complex systems, meticulous organization of and efficient accessibility to project-related information is crucial to project success.

With an increase in system complexity comes the need for effective knowledge management to promote efficiency and coordination in project teams. Information technology plays a key role in organizing, storing, and retrieving large amounts of knowledge and in allowing organizations to take advantage of the knowledge reuse paradigm [21]. However, knowledge management is more than simply storing documents in a searchable repository. It involves acquiring, sharing, and integrating knowledge from multiple perspectives into a shared understanding of a given problem and its intended solution [4].

Distributed cognition [40] stresses that the true power of human intelligence is captured only through the interaction of minds. Consequently, team members should not assume shared understanding of knowledge regarding design rationale, experience from previous projects, or task dependencies. To facilitate shared knowledge and synthesis of competing perspectives, distributed knowledge must be externalized and recorded, creating a physical record of the team's mental efforts in the form of a collective team memory [4]. A team memory should contain all knowledge related not only to the design product, such as design rationale, but also to the design process, such as team roles, responsibilities, contributions, and progress. This knowledge can be collected and maintained through the use of adequate communication and awareness mechanisms.

Team members need to maintain an overview of their project as a whole while ensuring that all members of the team have access to the same project-related knowledge. They need not only to remember how the design has evolved throughout the life of a project, but also to notice and understand recent changes that teammates have made

to the design. With this knowledge, team members should possess a better understanding of project tasks, dependencies, and risks as the design progresses and evolves.

By maintaining a prioritized list of design risks and by leveraging time-based visualization techniques similar to the SoftRisk visualization depicted in Figure 2.5, all members of a design team can see the current risk list as well as changes in the list over time. Additionally, each of these risks represents an individual mitigation task, which can be assigned to a member of the team. By incorporating this information into the risk visualization, the team can maintain an overview of the evolution of their design throughout the course of the project.

If a physical team memory is to be beneficial to project teams, it must be easy to maintain and use. The collection, organization, and archiving of project-related knowledge should be a natural by-product of the design process that adds minimal overhead to the project. Additionally, a team memory must be organized and presented to the team in such a way that team members can quickly notice and understand changes and potential problems and easily retrieve further details when necessary. Although the design and development of a complete team memory is beyond the scope of this thesis, the work contained within this document takes a step toward achieving this broader goal for project management. Development of a risk management model provides critical backend knowledge that can then be incorporated into a more elaborate team memory and visualized using time-based techniques. Research related to the development and implementation of this team memory is an important avenue for future work.

3.1.3 Guidelines for Management Tool Support

Effective project management requires a systematic process and supporting tools that add structure to the design process and facilitate team coordination. To the greatest extent possible, project management tasks must be incorporated into the design process with minimal added overhead. To accomplish these goals, tools to support project management in distributed design teams should adhere to four guidelines [51], which are outlined in Table 3.1.

The first guideline outlines the advantages of incorporating risk management into the design process. Prioritizing design risks promotes and structuring an iterative design process, helping designers to remain focused on the most critical design issues during each iteration. The remainder of this chapter elaborates on this guideline and the integration of a risk-driven management model into the SBD process.

3.2 Structuring Risk-Related Knowledge for Reuse

Many software projects suffer from common problems [10, 58], proving that teams do not effectively draw upon previous experiences and learn from their mistakes. Claims represent one form of capturing and archiving knowledge for retrieval and application within a new context. However, claims have thus far been limited to stating design features and tradeoffs in relation to a system interface, or *product*, being developed for an end-user. To aid project teams in learning from their management mistakes, claims must also serve to capture knowledge related to the *process* of designing a system.

Claims summarize particular aspects of design rationale, explicitly stating the positive and negative tradeoffs of a design feature. These tradeoffs represent the possible desirable and undesirable outcomes of incorporating a

particular feature into the design of a system. Thus, these tradeoffs explicate the uncertainty, or risk, associated with the design. Upsides correspond to opportunities while downsides represent risks. Constructing a claim is equivalent to identifying a collection of related project risks. The tradeoffs signify various effects caused by the design feature. Although claims are tied to a specific context of use, the underlying knowledge can be reused in subsequent projects. Consequently, designers can learn from their experiences in previous projects as opposed to creating each new interface from scratch.

1. Guide the design process with a risk-driven management model
Guiding design teams through the steps of a redesign process will promote iterative design. Prioritizing project risks draws attention to the key problems in the current design that should be addressed in the next iteration. Teams can quickly determine and allocate key tasks for redesign. Consequently, teams can remain focused on the most critical aspects of the project.
2. Support team coordination through activity awareness
Aiding distributed teams in the externalization and maintenance of a collective team memory will help team members to remain aware of the activities and perspectives of everyone on their team. A team memory should include knowledge related to progress, individual contributions, task assignments, decision rationale, and design evolution. The creation and maintenance of a team memory should be a natural by-product of the design process, adding minimal overhead to the project while helping teams to coordinate tasks and dependencies.
3. Organize project-related knowledge using time-based visualization techniques
Organizing an intuitive team memory will allow teams to monitor, reflect on, and improve their processes throughout the course of a project, while visualizing project-related knowledge according to time takes advantage of episodic memory. An effective activity timeline should allow team members to quickly understand design changes, notice potential problems, and retrieve more detailed information on demand. The activity timeline should help team members to maintain a "big picture" view of the project as it evolves over time.
4. Archive product and process-related knowledge for reuse
Maintaining a team memory throughout the life of a project and archiving product and process-related knowledge for reuse will allow future teams to find valuable knowledge and identify common mistakes early in the design process. A growing repository of both product and process-related knowledge contributes to the science of design.

Table 3.1: Guidelines for project management tool support, adapted from [46]

Based on the similarities between claims and project risks, the overall goal of this thesis is to adapt the use of existing, process-related risk management techniques for use in managing product-related risks within the context of a claims-centric, notification systems design. If the use of project claims to structure risk-related knowledge is proven beneficial to risk management, then the definition of a claim can be extended to include process-related knowledge. Section 3.2.1 discusses the management of product-related design risks and narrows the scope of the

problem to work accomplished in this thesis. Section 3.2.2 then discusses the need for an extension of the reuse paradigm and outlines the future steps needed to accomplish the broader goal of managing both product and process-related risks.

3.2.1 Managing Product-Related Design Risks

Existing risk management techniques, particularly within the software domain, typically focus on the management of process-related risks. Although process-related issues, such as project planning, are critical to the success of all professional projects, HCI as a discipline is primarily concerned with product-related concerns. The success of a system interface relies on the mitigation or elimination of critical design risks that might inhibit users' ability to accomplish their goals.

Even within the scope of product-related design claims, a simple solution for managing risk does not exist. The consideration of claim downsides as individual and independent project risks is a simplistic and idealistic view of HCI design. In reality, a number of complex interactions will cause problems at other levels of the design and the design process. For example, inaccurate system requirements might lead designers to comprise an entirely inaccurate set of claims. In this case, the design model defined for the system and user model might agree; however the interface will not meet the actual needs of the user. Additionally, the claims in a design set cannot be assumed as independent of one another. In contrast, the downsides included in a particular claim are typically related. For the purposes of this thesis and the simplicity of an initial risk management model, it is assumed that claims are primarily independent from one another, i.e. the occurrence of a downside in one particular claim has no effect on the occurrence of a downside in another claim. In reality, however, claim independence cannot always be assumed.

3.2.2 Extending the Reuse Paradigm

If designers can leverage knowledge from previous projects to improve their design product, they should also be able to leverage knowledge from previous projects to improve their design process. By reusing process-related knowledge from similar previous projects, designers can identify and mitigate common problems early in the project life cycle. Mitigation strategies can also be reused and customized to meet the needs of a specific project.

Basili's Experience Factory [7] was developed with these goals in mind. The system was devised to facilitate process improvement in software development by structuring, classifying, and storing packaged *experiences* from previous projects for reuse. Experiences include both product and process-related knowledge and are input into a repository as artifacts, examples, or lessons learned. Experiences are then tailored to meet the needs of a specific project and supplied on demand in the form of models, tools, or baselines.

Houdek [29] proposes the packaging of experiences as *quality patterns*, which contain both a problem and a solution within the context of a particular project. Quality patterns are classified according to their purpose, the viewpoint from which the experience was written, and the environment in which the experience took place. An example quality pattern, describing the experience of using a cleanroom method in software development, is depicted in Figure 3.1.

The concept of reusing process-related knowledge is a natural extension of the reuse paradigm; however, Basili's packaging of reusable experiences is too coarse. Reusing an experience is similar to reusing a generic

software process model that has been adapted for a specific project. In this way, attempting to reuse an experience is like attempting to reuse an entire claim set. Although this level of reuse might be possible, it is not a suitable starting point. A set of claims is tailored to the needs of a specific user and the design of a specific system, and each individual claim plays a role in defining the interface to be developed. The reuse of an entire claim set, originally created in the design of a different system, would require substantial rework to meet the needs of a new project.

Classification

type	process pattern
object	cleanroom method
purpose	impact on product quality
viewpoint	project manager
environment	laboratory experiment
analysis technique	GQM–based measurement program
restrictions	small commercial applications

Abstract Even with inexperienced developers the cleanroom approach produces software which fulfills the requirements more extensively and is better readable and maintainable than traditional developed software.

Problem Needed is a development method, first, which can be used by inexperienced programmers, second, produces software, which meets the requirements quite extensively, and third, can be easily maintained.

Solution The required properties can be achieved by applying the cleanroom method. The product quality is better in view of readability (more comments) and requirements conformance (significant higher than in traditional development). An application with inexperienced programmers is possible.

Context

a) Characterization of experiment:
⟨ Description of the cleanroom experiment ⟩
b) References to GQM model:
⟨ GQM goal, questions, and metrics as presented in fig. 4 ⟩

Example Cleanroom experiment

Explanation

⟨ collected data and interpretation as presented in fig. 6 ⟩

Related experience → Cleanroom method in large environments.

Administrative information

created at 15.07.1996 by Frank Houdek
changed at 18.02.1997

Figure 3.1: An example quality pattern, from [29] (c) 2005 ACM, Inc. Used by permission.

The same difficulty is true in the reuse of experiences, as too much information is stored within the smallest structure of knowledge. Instead, experiences must be decomposed into smaller chunks of knowledge appropriate for storage, retrieval, and reuse in a variety of projects across domains. One option for accomplishing this decomposition is to structure the knowledge contained within an experience as a claim. Product-related knowledge falls into the existing structure of a design knowledge claim. However, process-related knowledge can also be organized into a claim structure by decomposing projects into a set of process-related risks.

Houdek's quality patterns contain both a single problem and a single solution within the same packet of knowledge. In contrast, a claim represents a cause, coupled with a collection of related effects. Each of the negative effects has any number of possible solutions, each of which is encapsulated as a separate claim. These solution claims become mitigation strategies for resolving a particular risk and are related to the original claim through Wahid's mitigation relationship. Consequently, designers can identify a potential problem, or claim downside, and follow mitigation relationships to discover a number of alternative solutions. While examining these solutions, the designer will be alerted to other potential issues that might result from mitigating the original problem and can continue to follow relationships and mitigate downsides until satisfied with the result.

If structured properly, process-related claims can easily be reused. All projects have risks, and many risks occur consistently in projects across domains. Although the probability, priority, and impact of a particular risk will vary with each project, the general risk statement and possible mitigation strategies will be applicable to many different projects. Successful knowledge reuse relies on the appropriate definition of both product and process-related knowledge as well as an analysis of how the two types of knowledge can be juxtaposed to improve the design process. By decomposing projects into a set of product-related claims and a set of process-related claims and by treating the downsides of these claims as project risks, the claims analysis process begins to guide the design process in a new way and adds an element of risk management with minimal overhead. Moreover, both types of claims can be archived for reuse.

Consider, for example, the two claims in Figure 3.2. Claim A relates to the design product – a notification system. The two downsides of the claim represent potential problems that directly affect the quality of the product. Claim B, on the other hand, relates to the process by which the notification system is designed. The two downsides of this claim might also affect the quality of the resulting product, but in a more indirect manner. For example, if additional project resources were required but unattainable, then postponing a deadline could result in the cancellation of the project. A third claim, about accelerating the project schedule to stay on schedule for the next deadline, could be added to the set to mitigate the Claim B downside about the project falling behind schedule. However, the accelerated schedule might still require additional resources; therefore, a fourth claim would be needed to mitigate that risk.

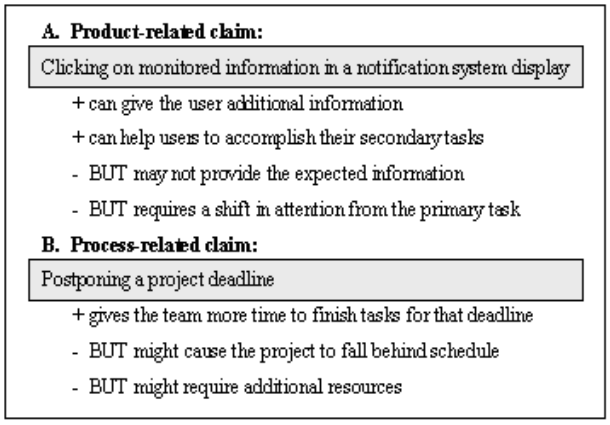

Figure 3.2: A comparison of product and process-related claims

An important first step toward improving both design product and process is to develop a risk-driven management model for the existing structure of a product-related design claim and to evaluate the contribution of injecting a risk management model into HCI design. The overall goal of this thesis is to initiate the use of existing, process-related risk management techniques for the management of product-related risks within the context of notification systems design. If the use of project claims to structure risk-related knowledge is shown to be beneficial to risk management, the definition of a claim can then be extended to include process-related knowledge.

3.3 Analyzing Design Risks

During each iteration of the design process, the underlying goal is to identify the most critical areas for improvement and to correct the design to document those changes. The new design is then re-evaluated to determine if the changes actually addressed the problems previously detected. Within SBD, a system design is documented as a set of scenarios and claims. The scenarios tie theoretical design knowledge, contained within the claims, to a particular context of use. Thus, the goal during each design iteration is to identify potential problems with the existing design, determine how the claim set should be modified to address those concerns, and then re-evaluate the resulting interface.

This section discusses key contributions of this work and outlines the development of a risk-driven, claims-centric for the management of product-related design concerns. The issues discussed in this section map directly the to key research questions addressed in Chapter 5. Section 3.3.1 outlines the advantages and disadvantages of managing design claims as opposed to individual downside risks. Sections 3.3.2 and 3.3.3 elaborate on the process of redesigning a system interface and discuss the benefits of risk management to SBD and the benefits of SBD to risk management. Section 3.3.4 returns to the concept of team coordination, mentioning how the success of this risk model and its future implementation within collaborative environments can improve project management. Finally, Section 3.3.5 discusses the various pieces of information that arise throughout the design process and can contribute to the estimation of a risk exposure value for each project claim.

3.3.1 Managing Risky Claims Versus Risky Downsides

Traditional risk management techniques could be applied to HCI design projects; however, the use of a claims-centric model aids in the integration of management tasks directly into the design process. Likewise, the incorporation of risk management into the SBD process serves to further the science of design and the science of software project management.

A turning point in the overall structure of this risk-driven model is the decision to prioritize claims as opposed to individual downsides. An individual downside represents a specific project risk, whereas a claim, as a whole, represents a cause and a set of associated effects with both positive and negative consequences. However, a claim, as a whole, is the fundamental packet of design knowledge, and an individual downside cannot meaningfully stand on its own outside of its associated claim. Although the prioritization of individual downsides most closely reflects the traditional concept of managing risk, prioritizing individual downsides also has the potential to magnify significantly

the number of project risks being prioritized. As the number of downsides grows large, it becomes more difficult to monitor risks throughout a project.

Roy's ProRisk framework [48] discusses the issue of managing a large number of project risks and suggests clustering related risks in an attempt to reduce the total number of project risks to prioritize. In the ProRisk model, risks related to similar aspects of a project are grouped together. These groups are then clustered again, using a tree structure, until all clusters can be brought together into one root project node. A claim, which connects a set of project risks that are caused by incorporating the same design feature into a system interface, is, in many ways, similar to Roy's risk clusters. A comparison of the ProRisk framework to the structure of a claim is presented in Figure 3.3.

This risk-driven model promotes the prioritization of project claims as opposed to individual downside risks. In doing so, the model maintains use of the fundamental packet of design knowledge while simplifying the prioritization process.

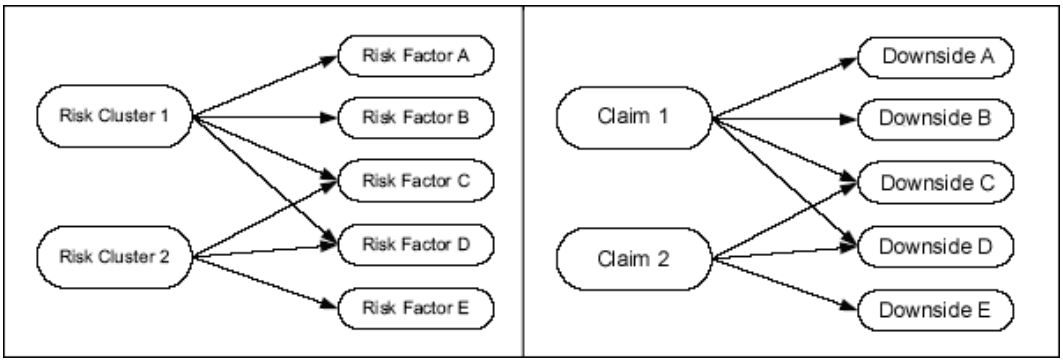

Figure 3.3: A comparison of the ProRisk framework to the structure of a claim

3.3.2 Risk Management in Support of SBD - Improving a Project Claim Set

Discrepancies between the design model and user model defined for an interface can be attributed to one or more failing, or "broken" claims. These claims can be thought of as interfering with the design goals as opposed to contributing to them. To improve the design of an interface, designers must identify and "fix" these failing claims.

A designer is typically alerted to usability problems through various forms of analytic and empirical evaluations. In interpreting the results of an evaluation, the designer must determine which claims in the system design are contributing the usability issues, and, as a result, interfering with overall design goals. The goal of the next design iteration is to "fix" the failing claims, by either mitigating troublesome downsides of those claims or removing those claims in favor of more appropriate design knowledge. Following a re-design phase, designers should evaluate the current interface to re-evaluate and compare the user and design models. The most critical project risks should be the focus of the evaluation so that designers can determine which risks have been mitigated, which downsides are still causing a problem in the interface, and which design concerns should be the focus of the next design iteration. However, given the inevitable limitations on time and project resources, designers cannot expect to address every design concern. Consequently, designers must prioritize their design concerns – i.e., assess

potential problems and determine which issues can and should be addressed within the projects constraints. A well-defined method and supporting tools to aid designers in prioritizing their design concerns could increase efficiency and productivity during redesign.

The integration of a risk-driven, claims-centric management model into the SBD process adds structure to the design process by guiding designers through a sequence of steps for redesigning their interface. As opposed to following an *ad hoc* approach to improving a system, designers can examine a dynamic prioritization of design risks and remain focused on the most critical issues for improvement during each design iteration. Consequently, designers can efficiently identify and assess design risks and address as many of the most critical concerns as possible within the constraints of a given project.

In addition to guiding the redesign process, use of the risk model also serves as a guide for evaluation. After addressing a subset of design issues during a particular iteration, designers should evaluate the effect of their design improvements before beginning the next iteration. A prioritized risk list not only determines the most critical design issues to address during the current iteration, but also the most critical design improvements to evaluate before the start of the next iteration. Once designers have determined which risks to evaluate, they can structure an analytic evaluation to acquire the necessary responses or develop an empirical test to produce the necessary results.

Aspects of the risk model fit into each stage of the SBD process. The next section describes the ways in which the key components of SBD contribute to the successful integration of risk management into HCI design.

3.3.3 SBD in Support of Risk Management – Facilitating Design Discussion

The goal of injecting risk management into HCI design is to increase the structure of the design process and improve the management of design projects without adding substantial overhead to the process. Consequently, knowledge is gathered at each stage of the design process and incorporated into the risk model. The key phases of the design process include requirements analysis, in which system requirements are elicited from the user, participatory negotiation, in which project stakeholders come together to discuss their needs for the system, and a number of design iterations, during which an initial system image is created and refined. In between the design iterations, analytic and empirical evaluations are conducted to test the design and discover existing problems.

During the requirements analysis phase, designers determine the tasks to be accomplished by the system and develop a set of scenarios and claims depicting the users' existing problems. Based on the needs of the user, an initial system image is created, including the design model IRC rating for the system and a set of scenarios and claims depicting the design choices made during activity, information, and interaction design.

During a participatory design session, stakeholders examine the design and rate the importance of each design tradeoff. Since these stakeholders often have backgrounds in very different areas, it can be difficult for them to communicate effectively and accomplish their goals in a short period of time. The use of natural language scenarios and claims helps stakeholders to communicate their ideas and bring various perspectives to bear on the design of the interface. Scenarios provide a concrete context in which to consider use of the system, while claims draw out the key features of each scenario and explicate the design tradeoffs that will affect each stakeholder's satisfaction with the delivered system.

An analytic evaluator provides his (or he) opinion of the design, and users evaluate the system to determine the user model IRC rating and the effect that particular tradeoffs have on the current design of the interface. The use of claims aids designers in determining specific data based on particular design tradeoffs that should be collected during these evaluations. The knowledge collected and stored as claims throughout the SBD process allows the designer to estimate design risk early and continually improve that assessment throughout multiple design iterations. Additionally, the risk-related knowledge stored within the structure of a claim can be reused in subsequent projects.

3.3.4 Toward Supporting Team Coordination

Integration of a risk management model within the design process explicitly addresses the first guideline for providing management support. However, it indirectly contributes to an implementation of the remaining three guidelines. Once a risk management model is in place, adequate tools can help to improve team coordination, knowledge management, and knowledge reuse. In this way, the work contained within this thesis lays a solid foundation for developing more effective tools and techniques for the management of collaborative teams.

When a risk management model is supported by tool technologies, a prioritized list of claims can be developed early in a project and updated automatically as claims are added and removed from the project and as more information becomes available. To maintain a team memory of the project, changes in claim priority should be documented and displayed using time-based visualization, such as the visualization used in SoftRisk [32]. With a visual record of design evolution available to all team members, the team is better equipped to monitor progress.

Prioritization of project claims draws attention to key downsides that should be considered for mitigation. Each downside chosen for mitigation identifies one or more project tasks that can be allocated to specific team members. As downsides are mitigated, i.e., as new claims with mitigating relationships are added to the project set, those tasks are completed. Consequently, other members of the team can monitor progress of downside mitigation. Support for individual task allocation and the maintenance of a time-based team memory aids project teams not only in monitoring individual and team progress, but also in maintaining an overview of the project as it changes over time.

3.3.5 Estimating Risk Exposure

A design concern is exemplified by the tradeoffs contained within a claim. Thus, by prioritizing the claims in a design set, designers can determine which design concerns are most critical at any given time and remain focused on those most critical concerns throughout the redesign effort. The prioritization of claims requires the calculation of a risk exposure value for each claim, based on a combination of probability and impact values for one or more relevant factors. A number of factors warrant consideration for inclusion in an initial risk-driven management model. The goal of risk management is to make an initial risk assessment as early as possible and to continually refine that assessment as more information becomes available throughout the course of the project. The cost of resolving a risk early in a project is often relatively low; however, that cost increases significantly over time. Given the volatility of a project during its early stages, accurate measurement of progress, correctness, and design stability is difficult. However, risk management that begins early in the development cycle and continues throughout is critical to project success. Any information available early in the project, whether the information is related to the current project or to similar previous projects, should be utilized in an early risk assessment.

The fundamental estimation of probability and impact for a project risk still applies in this claims-centric approach. However, as opposed to calculating a single estimate for each of the two factors, the claims-centric model seeks to reduce overhead by extracting information from various stages of the design process and factoring that information into the risk calculations. As with other risk models, the expectation is that the accuracy of the estimates will improve as more information becomes available.

To incorporate knowledge from multiple perspectives, the model considers data obtained from project stakeholders and relevant knowledge from previous projects, as well as the results of internal evaluations as that information becomes available. Use of multiple factors helps to correct individual bias and ensure that the maximum amount of knowledge is utilized. Additionally, it prevents any one individual from possessing the power or the burden of managing risk for the entire project team. To further prevent biased and inaccurate results, the claims-centric model incorporates objective measures, such as IRC ratings and results from empirical evaluations, whenever possible. When subjective measures are necessary, the rating scales are defined to be as objective as possible. Additionally, the model considers the quality of any evaluator who provides a subjective response.

In each design phase, claim exposure is determined by the following factors:

- **Prior to the performance of any internal evaluation:**
  Claim risk exposure is calculated as a combination of:
  1. Base claim quality rating
  2. Difference in design model and claim IRC values
  3. Amount and quality of relevant external rationale
  4. Degree of downside mitigation
  5. Stakeholder concern rating
- **Following an analytic evaluation, but prior to an empirical evaluation:**
  Claim risk exposure is calculated as a combination of factors 1 through 5 above, plus:
  1. Opinion of HCI expert in terms of the degree to which a problem exists
  2. Degree of confidence that the evaluator's assessment is correct
- **Following an empirical evaluation:**
  Claim risk exposure is calculated as a combination of factors 1 through 5 above, plus:
  1. Degree to which user evaluation shows existence of a problem
  2. Confidence in the accuracy of the experiment

Knowledge is gathered at the level of both the claim and the individual downside. Specific factors considered at each level are outlined in the remainder of this section.

3.3.5.1 Knowledge Based on Individual Claims

At the claim level, the risk impact factor should consider the degree to which a particular claim contributes to the design goals. This consideration can be quantified through a comparison of the design model IRC rating defined for the system to the IRC rating assigned to a particular claim. This difference in IRC ratings represents the impact

factor of the risk equation. Additionally, the risk probability factor should take into account the overall quality of a given claim. A high quality claim is more likely to contain valid design knowledge, while a low quality claim should be considered risky regardless of the situation to which it is applied.

Use of this IRC difference factor relies on prior definition of critical parameters for a given design domain. Since critical parameters have been defined for notification systems, this work focuses on developing a risk-driven management model tailored specifically for notification systems design. However, after defining critical parameters, this difference factor could easily be redefined for a new design domain.

The optimal definition of claim quality is arguable; however, key contributing factors include the quality of supporting design rationale, the experience of the claim author, and the degree to which a claim is reused in the design of multiple systems. An initial algorithm for calculating claim quality is explained, along with the rest of the risk model calculations, in the next chapter.

3.3.5.2 Knowledge Based on Individual Downsides

At the individual downside level, the risk impact factor should consider evidence of a sizeable downside effect, cited from both external rationale and the results of internal evaluations. Internal evaluations refer to testing, either analytic or empirical in nature, which is performed on the actual interface being designed. The risk probability factor should consider the designer's confidence in each evaluation or piece of rationale. Confidence in internal evaluations might consider the experience of an analytic evaluator or the statistical confidence in the results of an empirical evaluation. Confidence in external rationale should consider the quality of the source of rationale as well as its relevance to the current design domain.

External rationale refers to published work in related areas of research that provides insight into the effects of a particular downside as found in similar systems. The relevance of this rationale must be determined according to a valid classification schema. For example, in relation to the design of a large-screen information exhibit, rationale cited from work with other large-screen displays will be more relevant to the current design than work involving handheld devices.

The factors considered in determining relevance must be customized to each design domain. In the design of notification systems, a number of different factors could be considered in determining the relevance of a piece of rationale; however the overall goal is to capture the essence of a notification system with a relatively small number of factors. Key issues to consider include the size and use of the interface being described. Three factors chosen for consideration are outlined in Table 3.2.

The System Accessibility factor refers to the use of the display within a physical space. A system might be placed in a public area, such as a hallway or the lobby of a building, in a semi-public area, such as an office or laboratory, or in a private area, such as a private office or cubicle. The number of people who could potentially interact with the system plays a role in determining what tasks can be accomplished with the interface, how information is displayed, and how users can and should interact with the system. For example, an interface for private office use can easily be approached by one user, who can then physically interact with the system by pressing buttons or selecting menu items. However, a system used by hundreds of individuals in the lobby of a large building must require far less physical interaction to meet the needs of its large user base.

Relevance factor	Description	Options
System accessibility	Where will the interface be physically located for use?	Public Semi-public Private
Display size	How large is the physical interface?	Large-scale Desktop-scale Handheld-scale
Use of display	Where will the interface be placed in relation to the user's primary task?	Primary display Secondary display

Table 3.2: Factors considered in determining the relevance of external sources of rationale

The Display Size factor takes into account the size of the interface. The system might be large-scale, such as a large-screen display or ambient exhibit, desktop- scale, such as a traditional desktop display or a medium-scale off-the-desktop device (for example, an in-vehicle navigational system), or handheld-scale, such as a PDA or other small-scale mobile device. The size of the device dictates the number of pixels with which to display information and also plays a role in determining the tasks that can be accomplished and the ways in which the user can interact with the device.

Finally, the Use of Display factor determines where and how the interface is used in relation to the user's primary task. An interface that is placed on the user's primary display must compete with the user's primary task not only for attention, but also for pixels on a single display. Primary display notification systems are typically small; however, they are often more likely to interrupt the user's primary task. In contrast, an interface placed on a secondary display can encompass more pixels without taking screen space away from the primary task. Consequently, secondary display notification systems can more easily remain in the user's periphery.

While external rationale gathers relevant information from previous similar project, internal evaluation refers to either an analytic or empirical test performed on the actual interface being designed. For an analytic evaluation, the calculation of an impact factor might take into consideration the opinion of the evaluator in terms of the degree to which the effect of a particular downside is evident in the current design of the interface. The probability factor must consider the evaluator's degree of confidence in the accuracy of his (or her) assessment, for example, by taking into account the evaluator's level of experience with respect to HCI theory and design work.

For an empirical evaluation, numerical data from a statistically significant experiment can be used to determine the degree to which the effect of a downside is evident in the interface and the degree of confidence in the accuracy of the evaluation. Factors to consider include the differences detected among experimental groups, the size of each user sample, and the confidence level to which the experiment can be shown to be statistically significant.

Although evidence of the effect of a particular downside, detected through relevant external rationale and internal evaluations, is a critical factor in determining risk, it must be considered in conjunction with the needs of the user. A downside might be determined to have a significant effect on the interface being designed; however, if the effect of that downside is of little concern to the project stakeholders, then that particular risk should not achieve high priority. Consequently, a stakeholder concern rating for each downside must also be taken into account when assigning risk exposure. Ideally, these concern ratings would be discussed and agreed upon by a number of project

stakeholders during a participatory design session. If such a session is not possible, concern ratings can be collected from multiple stakeholders and averaged or assigned by the project design team on behalf of project stakeholders.

Finally, the degree to which a particular downside has been mitigated must also be considered when calculating risk. The probability of the occurrence of a risk event depends in large part on the depth and implementation of a mitigation strategy. Although a risk might never completely disappear, the likelihood of it causing a problem in the design might decrease dramatically if all possible measures are taken to mitigate the risk.

3.4 Adapting Risk Management for Scenario-Based Design

The overall vision that inspired this work is to utilize the structure of a claim in managing both process and product-related design risks. This chapter establishes a comparison between claims and project risks and discusses the importance of structuring claims to store not only product-related design knowledge, but also process-related knowledge that could be reused from project to project. However, the sequence of steps required to accomplish this broad goal of integrating management support for both product and process-related knowledge is beyond the scope of this thesis. Thus, this chapter also narrows the scope of this work to address the management of product-related knowledge within the context of HCI design, taking advantage of the existing structure of reusable design knowledge in the form of claims.

By automating the risk management process and implementing an adequate risk model within a collaborative design environment, prioritization of project claims can not only guide design improvements, but also improve the management of team projects. A dynamic, prioritized list of project claims aids designers in identifying and focusing on critical design issues. By automating this process, risk management takes place within the design process, adding minimal overhead to the project. The goal of risk management in other domains is to make visible the most critical project risks by drawing those risks to the top of a prioritized list. Likewise, the goal of a risk model for HCI design is not to provide the designer with a perfectly ordered list of issues to address, but, instead, to flag potential problems that should be given additional consideration during the next design iteration.

The final sections of this chapter introduce the key components considered in the development of a risk-driven, claims-centric management model for integration within the SBD process. The next chapter presents the details of two risk models –one comprehensive model, which includes the factors discussed previously, and one simplified model, which includes only a rating of stakeholder concern for the effects of project downsides. The results of both models are then compared in Chapter 5.

Chapter 4 : A Risk-Driven Management Model for Scenario-Based Design

The purpose of developing a risk model for HCI design is to provide a well-defined method of assessing project risk and to integrate that assessment directly into the SBD process to achieve maximum benefit while minimizing project overhead. Chapter 3 outlined a number of critical factors needed to map important HCI concepts onto proven risk management paradigms. However, there is a tradeoff to consider in developing a beneficial risk model. The accuracy of risk predictions typically increases as more information becomes available and is incorporated into a model. However, the cost of a more comprehensive model is more complex calculations that might leave designers unsure of the risk assessment process and untrusting of the results.

To determine which type of model is more accurate, useful, and trustworthy, the details of two separate models are described here and evaluated in Chapter 5. The first model, presented in Section 4.1, is a simplified model taking into account only one risk factor –the degree to which project stakeholders are concerned about the effects of individual project downsides. The calculation of a risk exposure value for each project claim is, in this case, a simple average of downside stakeholder concern ratings. The second model, presented in Section 4.2, is a comprehensive model, including not only the stakeholder concern factor, but also a number of other factors discussed in Chapter 3. The calculations for this inclusive model are more complex; however, the high-level components considered in the model are more important than the specific calculations of the underlying sub-components. These low-level calculations serve as a proof of concept for the risk model as opposed to a proven set of equations for calculating design risk. Both the stakeholder concern model and the comprehensive risk model have been implemented for use within the LINK-UP system. The implementation details are presented in Section 4.3.

4.1. Stakeholder Concern Model

The stakeholder concern risk model prioritizes claims according to their degree of risk exposure, with a high exposure value resulting in a higher priority rating. As its name suggests, the stakeholder model considers only one factor in the calculation of claim risk exposure. This stakeholder concern rating (SC) represents the degree to which project stakeholders are concerned with the possible effect of each project downside. In this way, it rates the importance of a claim downside, based on the goals of the end-user and other project stakeholders.

Stakeholder concern ratings measure the degree to which stakeholders are willing to accept the effect of a particular downside. A value will be assigned to each downside along a 4-point decimal scale with a minimum of 1.0 and a maximum of 4.0, based on the point indicators presented in Table 4.1. A stakeholder concern rating of 1.0, representing a downside effect that is irrelevant to the goals of the system's end-users, has no effect on the overall

exposure of that downside. In contrast, a stakeholder rating of 4.0 represents a downside effect with critical negatives consequences for the end-user.

Stakeholder concern rating point indicators	
1.0	The effect of this downside will not interfere with my goals, mitigate opportunistically
2.0	Maintaining the upsides of this claim is more important than mitigating this downside, mitigate only if it is convenient to do so
3.0	Mitigation of this downside is important and should be given priority
4.0	Mitigation or elimination of this downside is absolutely essential to the success of the interface

Table 4.1: Point indicators for use in assigning stakeholder concern ratings

Risk management in other domains is often considered a task for the project manager; however, a team of project stakeholders can improve the quality of a risk assessment by bringing diverse knowledge and perspectives to the decision-making process and by correcting the biases created by an individual's specialized role within a project. A recommended process for group risk assessment is to first elicit individual knowledge from the participants, then share, combine, and integrate this knowledge through group discussion [27]. Ideally, stakeholder concern values would be determined by a group of project stakeholders during a participatory design session. However, if a participatory design session is not possible, then the system designers can rate the downsides on behalf of the project stakeholders, based on the system requirements.

Although a number of individuals can bring different stakeholder concern ratings from varying perspectives, only one group value will be input into the model for each downside risk. However, the procedure for assigning a single concern rating to each downside can be customized to meet the needs of a specific project group. For example, the individual stakeholders' concern ratings could be averaged to produce a single rating for the model. For a more complex model, each stakeholder could be assigned an importance value, which can then be used along with the individual concern ratings to produce a weighted average. Regardless of the technique used to determine the stakeholder concern rating for each downside, only one value needs to be input into the model.

Once a stakeholder concern value has been assigned to each downside within a claim, these downside values are averaged according to Equation 4.1.1 to produce a claim exposure value (CE) for that claim.

$$CE = \frac{\sum DSE}{\#downsides} \qquad \textbf{Equation 4.1.1}$$

Claims are then prioritized according to claim exposure value, with a higher CE value resulting in higher priority.

4.2. Comprehensive Risk Model

In contrast to the single-factor model presented in Section 4.1, this section presents a second, inclusive risk model that considers a number of risk factors. The two models will be compared interms of usefulness and accuracy in a user evaluation described in Chapter 5.

The comprehensive risk model also prioritizes claims according to their degree of risk exposure, with a high exposure value resulting in a higher priority rating. The claim risk exposure calculation involves a number of factors, depending on the current stage of the design project and thus, the amount of information available. Information is collected from the claim author during claim creation, from other designers who reuse the claim in their own projects, from project stakeholders during participatory design, and from system evaluators during either an analytic of empirical evaluation of the interface. The composition of the model is outlined in relation to each stage of the design process in Figure 4.1.

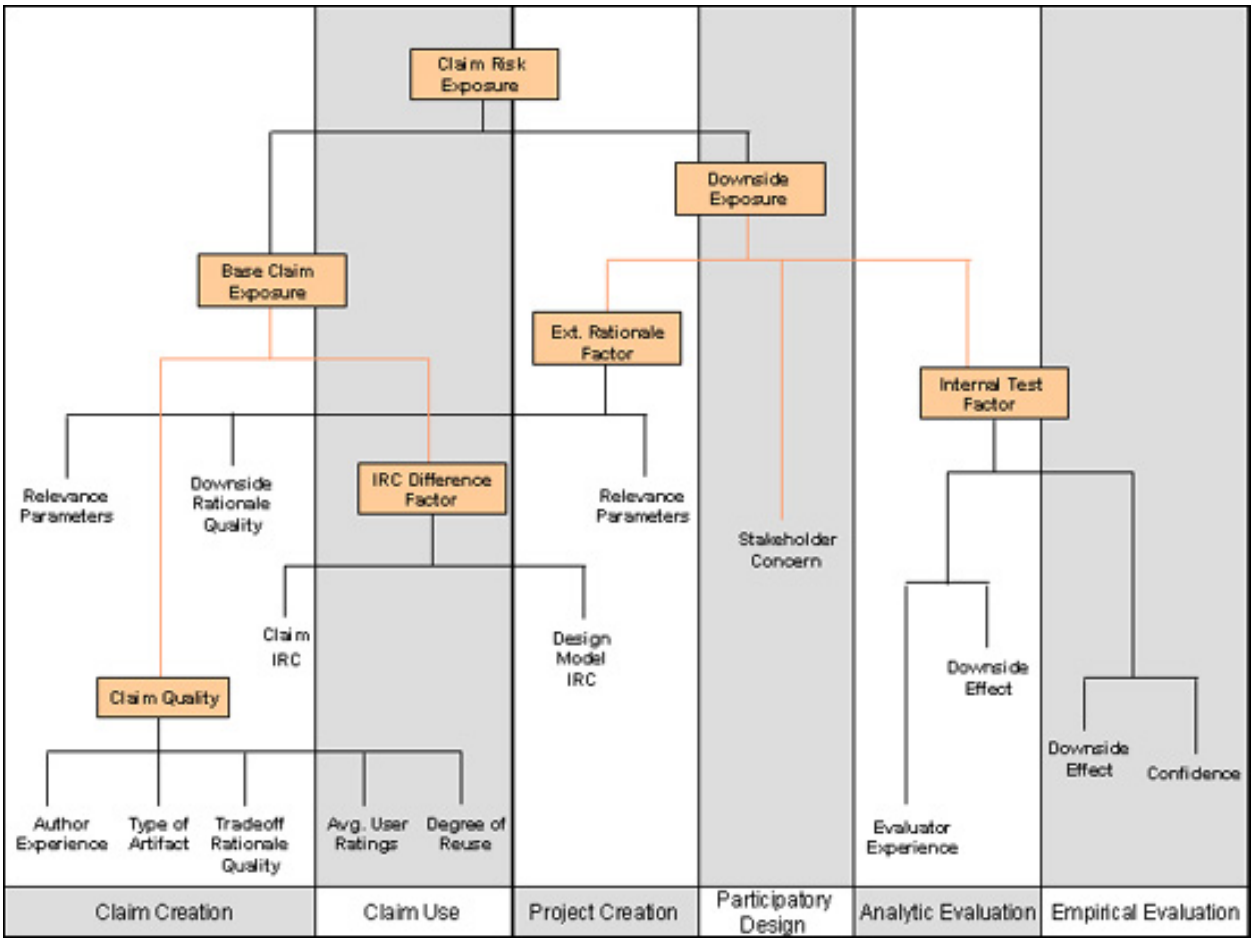

Figure 4.1: Hierarchical overview of the comprehensive risk model

In this hierarchical diagram, the horizontal position of the elements in the lowest tier of the hierarchy depicts the design phase during which that piece of information becomes available for use in the risk model. The elements surrounded by orange boxes represent compound factors, composed of the corresponding items in next tier of the hierarchy. These high-level elements are the most critical aspects of the model, representing key factors in the calculation of design risk exposure. The low-level components comprise an initial elaboration of the risk model, representing a proof of concept for use in evaluating the usefulness of prioritizing design claims and in determining

the most critical factors for the estimation of design risk. These equations are not a flawless model for managing design risk, but rather an initial mapping from important HCI concepts to key elements for estimating risk.

Overall claim risk exposure (CRE) is calculated according to Equation 4.2.1, as a combination of a base claim exposure value (BCE) and the maximum downside exposure value (DSE), taken over all downsides within a claim.

$$CRE = BCE + \max_{claim}\{DSE\} \qquad \textbf{Equation 4.2.1}$$

The base claim exposure value ranges from [0 - 1], and the downside exposure value ranges from [0 – 4]. However, the downside exposure value will only equal zero when no downsides exist for the claim. Thus, the overall claim exposure value will range from [0 – 5].

4.2.1 Base Claim Exposure

The base claim exposure value (BCE) considers the base claim quality rating (CQ) as well as the IRC difference factor (IRCD). The base claim exposure value accounts for one fifth of the total claim exposure. Since the claim quality and claim IRC values are stored with the claim and transferred along with the claim to every project that includes it, these factors provide little insight into the characteristics of a particular project, and, thus, are given less weight in the overall equation.

However, the IRC difference factor and base claim quality rating do provided information worth including in the model. As discussed in Chapter 2, a notification system design is assigned a design model IRC rating based on the overall goals of the system. Each claim is also assigned an IRC rating, depending on its anticipated or observed effect in relation to each of the here critical parameters. Although the exact nature of the relationship between the IRC ratings of a set of claims and the design model IRC rating for the overall system is an open question that remains under debate, it is generally accepted that a claim with an IRC rating similar to that of the design model will be better suited to contribute to the goals of the system. Consequently, the difference between the design model IRC rating and an individual claim IRC rating provides an indication of the degree to which the claim contributes to or interferes with a system's design goals. In this way, the IRC difference factor (IRCD) represents an initial measurement of the magnitude of a design risk.

The base claim quality rating (CQ) accounts for the idea that a given claim, if of questionable quality, is a risky addition to any system design. When combined with the IRC difference factor, the claim quality factor provides an indication of the designer's confidence in the accuracy of the claim's IRC rating. Given a low quality claim, the designer may question the validity of the claim's IRC value. However, given a high quality claim, the designer can be fairly confident that the reported claim IRC rating is accurate. As a result, the base claim quality rating represents an initial measurement of the probability that the given claim will, in fact, cause a problem in the design. The components considered in determining claim quality and IRC difference, along with the precise methods of calculating these two values, are presented in the following sections.

4.2.1.1 Claim Quality

The claim quality factor takes into account the idea that some claims present a fundamental risk, regardless of the project to which they are applied. These risky claims may be poorly constructed, supported by inadequate rationale, or written by novice designers. None of these factors, when considered in isolation, can accurately determine the overall quality of a claim. However, a number of factors in combination increase the likelihood that a specific piece of knowledge is of questionable quality. Consequently, claim quality (CQ) is based on a weighted average of five factors:

1. Average quality of downside rationale (RQ)
2. Author experience (AE)
3. Type of associated artifact (AT)
4. Degree of reuse (DR)
5. Average user rating (UR)

Each of the five factors is assigned a value on a [1 – 4] scale. The factors are described in more detail below. Each of the five components is assigned a weighting factor of 10, 20, or 30 percent of the overall claim quality rating. Higher percentage weights are assigned to those factors that have the capacity to change over time as the claim is reused in multiple design projects.

The first factor, rational quality (RQ) represents the average quality of claim tradeoffs, where the quality of a tradeoff is determined by the quality of the source of external rationale for a specific upside or downside. To preserve the overall [1 – 4] clam quality rating scale, if a claim contains no upsides or downsides, then the rational quality value defaults to a value of 1.0.

The quality of a cited source of rationale is determined by two factors: the venue in which the work was published and the quality of the results reported in the publication. For each claim upside and downside, the claim author will choose the most appropriate option for each of two factors depicted in Table 4.2. Consequently, a high-quality claim will contain design rationale from work publishing empirical results in journal papers. A source of

Rationale quality factors	
Venue: This source of rationale was published in the following type of venue:	
1	Not published
2	Short conference paper or poster
3	Full conference paper
4	Journal paper
Results: This source of rationale contains the following type of results:	
1	No evaluation reported
2	Results of an analytic evaluation reported
3	Results of a small-scale (sample size 12 or less) empirical evaluation reported
4	Results of a large-scale (sample size more than 12) empirical evaluation reported

Table 4.2: Factors contributing to the quality of sources of rationale

rationale that was not published, or that reported no evaluation results, is considered poor quality. The [1-4] scales representing these two factors are initial versions only. Determination of the adequacy of individual publication venues and the significance of evaluation results are complex issues that must be reassessed in the improvement of these initial claim quality calculations.

If more than one option for venue or results applies to a given source of rationale, then the highest selected value will be input into the model. The two factors are averaged according to Equation 4.2.2.

$$Factor\_avg = \frac{Venue + \mathrm{Re}sults}{2} \qquad \textbf{Equation 4.2.2}$$

This value ranges from 1.0 to 4.0, with a value of 1.0 representing poor quality rationale that was not published and did not report significant evaluation results and a value of 4.0 representing high quality rationale that was published in a journal and reported the results of a significant, large-scale empirical study.

After computing this average for each of the claim's tradeoffs, a sum of the factor averages is taken over all claim tradeoffs. This value is then divided by the number of tradeoffs, according to Equation 4.2.3, to produce the overall rationale quality rating.

$$RQ = \frac{sum(Factor\_avg)}{Num\_tradeoffs} \qquad \textbf{Equation 4.2.3}$$

Quality of rationale is an important, yet mostly static, factor in determining claim quality and thus accounts for 20 percent of the overall claim quality rating.

The second factor, author experience (AE), quantifies the claim author's level of experience according to two factors: knowledge of HCI and design theory and experience in designing notification systems applications. The system administrator will assign an author experience rating to each registered user of LINK-UP based on the two factors presented in Table 4.3. Knowledge of HCI theory measures the degree to which the claim author has been exposed to HCI research and the resulting literature. Design experience measures the degree to which the author has participated in the design of user interfaces. The author experience rating is calculated as an average of the two factors, according to Equation 4.2.4.

$$AE = \frac{Theory + Application}{2} \qquad \textbf{Equation 4.2.4}$$

This value ranges from 1.0 to 4.0, with a value of 1.0 representing a new designer with no design experience or knowledge of HCI theory and a value of 4.0 representing a knowledgeable and experienced HCI designer. Since it is possible for a novice designer with knowledge of SBD to create a better claim than an experienced designer with no knowledge of SBD, the author experience factor accounts for only 10 percent of the overall claim quality rating.

Author experience factors	
Knowledge of HCI and design theory	
1	No knowledge
2	Novice knowledge
3	Intermediate knowledge
4	Expert knowledge
Experience designing notification systems applications	
1	No experience
2	Novice experience
3	Moderate experience
4	Advanced experience

Table 4.3: Factors contributing to the author experience rating applied in the calculation of claim quality

The third factor accounts for the type of artifact (AT) associated with a given claim. The claim author assigns the most appropriate option, according to the scale depicted in Table 4.4, when updating an artifact into the LINK-UP system.

Type of artifact	
1	No artifact uploaded
2	One or more screenshots depicting system features
3	Animated movie depicting system functionality
4	Interactive prototype depicting key system functionality

Table 4.4: Point indicators for determining type of artifact associated with a claim

The type of artifact associated with a claim is typically a static value that provides insight into the amount of time and effort devoted to creating an evaluating a claim. However, a claim with no associated artifact is not necessarily a poor quality piece of knowledge. Thus, this factor accounts for only 10 percent of the overall claim quality rating.

The fourth factor, degree of reuse (DR), is determined by the number of projects in which a given claim has been used. This value is maintained automatically in LINK-UP according to the scale depicted in Table 4.5. This scale considers the idea that a claim's quality will be recognized by a number of different designers. Thus, the degree of reuse value will increase relatively quickly as the claim is reused in a handful of projects. Once the claim has been reused in 10 different projects, it will receive the maximum quality rating for this particular factor. Since this factor has the potential to increase over time, degree of reuse accounts for 30 percent of the overall claim quality rating

Number of projects reusing the claim	
1	0 or 1 projects
2	2 to 5 projects
3	6 to 9 projects
4	10+ projects

Table 4.5: Point indicators depicting the degree to which a particular claim has been reused

The final factor considered in calculating claim quality is an average of user ratings for a given claim. Registered users of LINK-UP have the option to rate claims with which they have experience. Although the user rating is entirely subjective, users are encouraged to consider quality factors, such as readability, comprehensiveness, and usefulness, when rating individual claims. Designers can rate claims according to the scale depicted in Table 4.6.

Claim user rating	
1	Poor
2	Below average
3	Average
4	Above average
5	Excellent

Table 4.6: Point indicators used in assigning a user rating for a given claim

All use ratings for a given claim are then summed and divided by the absolute number of ratings to produce an average, as shown Equation 4.2.5.

$$Avg\_rating = \frac{sum(User\_ratings)}{\#ratings} \qquad \textbf{Equation 4.2.5}$$

To preserve the overall [1 – 4] scale, the number of users who rate a claim is converted according to Table 4.7.

Number of users who rated this claim	
1	0 or 1 users
2	2 to 5 users
3	6 to 9 users
4	10+ users

Table 4.7: Point indicators depicting the number of users who have rated a particular claim

Finally, the author experience (AE) value is factored into the equation, accounting for the idea that the ratings of experienced users should weigh more heavily on the overall average because experienced users possess a better understanding of what constitutes a high quality claim. Thus, each level of author experience is assigned a percentage of a user to be used in calculating the Num_ratings factor, as follows:

For each user who rated the claim:
- if AE = 1 then Num_ratings += 0.25
- if 1 < AE <= 2 then Num_ratings += 0.50
- if 2 < AE <= 3 then Num_ratings += 0.75
- if 3 < AE <= 4 then Num_ratings += 1.00

The final value of Num-ratings is then mapped to the [1 – 4] number of users scale defined above. The final average of all user ratings (UR) is calculated according to Equation 4.2.6.

$$UR = Num\_ratings^{\frac{Avg\_rating}{5}}$$ **Equation 4.2.6**

In this equation, the Num_ratings factor refers to the relative [1 – 4] scale defined above. Average user rating also has the potential to increase or decrease over time; thus, it accounts for 30 percent of the overall claim quality rating.

After these five factors have been determined, the overall claim quality rating is calculated on a [1 – 4] scale according to Equation 4.2.7.

$$CQ = \frac{(2*RQ)+(AE)+(AT)+(3*DR)+(3*UR)}{10}$$ **Equation 4.2.7**

The base claim quality rating is maintained with the claim at the library level. The value can change over time as the claim is reused in more projects and as more users rate the quality of the claim.

4.2.1.2 IRC Difference Factor

The IRC difference value (IRCD) calculates the difference in IRC parameter values between the design model and the claim in question. In this calculation, the difference is treated as the distance between two three dimensional points. The value ranges from [0 – 1] and is calculated according to Equation 4.2.8.

$$IRCD = \frac{\sqrt{(a_I - b_I)^2 + (a_R - b_R)^2 + (a_C - b_C)^2}}{\sqrt{3}}$$ **Equation 4.2.8**

A claim with an IRC rating similar to the design model IRC rating for a given system is more likely to contribute positively to the goals of the design than a claim with a wildly different IRC rating. Considering the three

dimensional IRC cube, the maximum distance between any two points is equal to $\sqrt{3}$. To maintain the [0 – 1] scale for the IRC difference factor, the calculated distance is divided by this maximum value.

After the claim quality (CQ) and IRC difference (IRCD) factors have been determined, the base claim exposure value (BCE) is calculated according to Equation 4.2.9.

$$BCE = IRCD^{(\frac{CQ}{4})} \qquad \textbf{Equation 4.2.9}$$

In this equation, the base claim exposure value (BCE) ranges from [0 – 1]. Thus, the base claim exposure value increases as the IRC difference value increases. For a given difference in IRC, the claim exposure value will decrease as claim quality increases. However, as IRC difference increases, this decrease due to claim quality becomes smaller. An IRC difference of 1.0 achieves a base claim exposure value of 1.0 regardless of the quality of the claim.

According to this equation, if the IRC difference factor equals zero, then BCE will also equal zero. However, given a low quality claim, the accuracy of this IRC value is unknown, and the claim should be considered a risk. To prevent this misleading exposure value, an exception is added to the equation. If the IRC difference factor equals zero then BCE = (1 – CQ/4). The base claim exposure value (BCE) ranges from [0 - 1]. The range of values for claim quality, IRC difference, and the resulting base claim exposure are visualized in Figure 4.2.

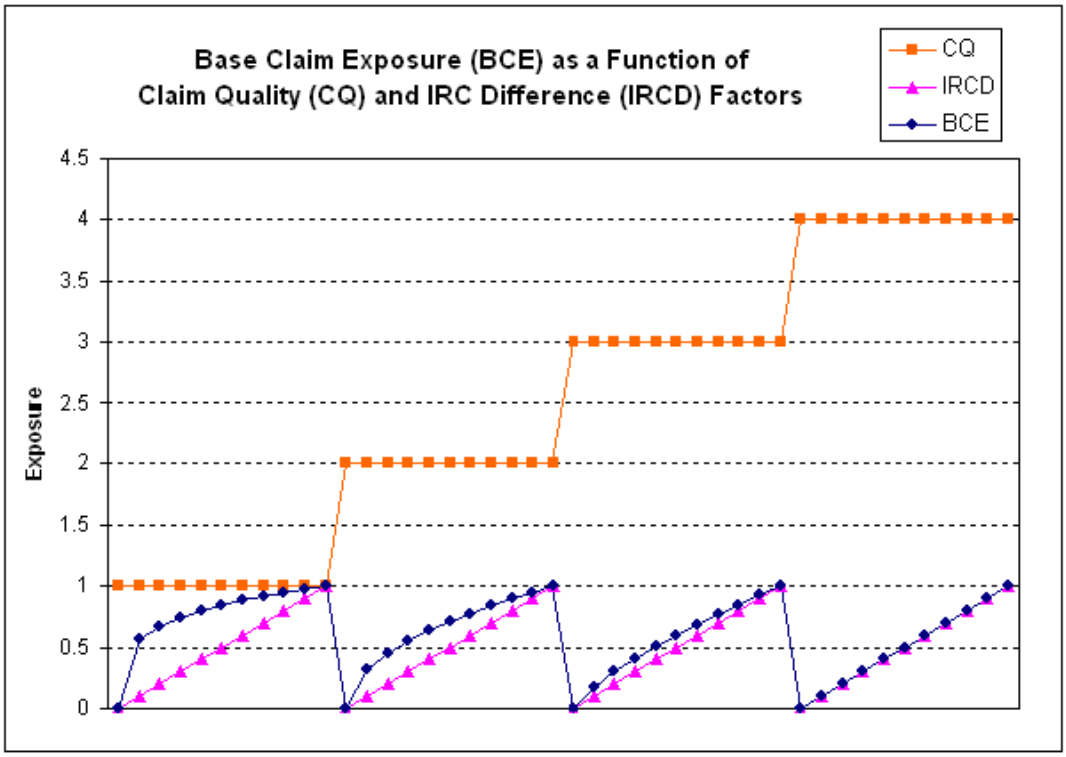

Figure 4.2: A visual depiction of Base Claim Exposure as a function of Claim Quality and IRC Difference

4.2.2 Downside Exposure

Each downside of a claim is assigned a downside exposure value (DSE), and the maximum exposure value over all downsides of a particular claim is factored into the overall claim exposure equation. The use of one maximum downside exposure value, as opposed to a summation of the exposure values for all downsides in a given claim, ensures that claims with one critical downside are not outweighed by claims with multiple non-critical downsides. Thus, the most critical downside of a given claim will primarily determine the priority of that claim. This method of calculation maintains the simplicity of prioritizing claims while preserving the significance of prioritizing individual downside risks.

The downside exposure value (DSE) accounts for four key factors:

1. Stakeholder concern rating (SC)
2. External rationale factor (R)
3. Internal testing factor (T)
4. Mitigation factor (M)

Each of these factors is defined in more detail below. The downside exposure value (DSE) is then calculated according to Equation 4.2.10.

$$DSE = \left[\frac{SC(\frac{1}{R} + 2T)}{3} \right] * M \qquad \textbf{Equation 4.2.10}$$

The stakeholder concern rating (SC) is defined on a [1 – 4] scale, with these values being assigned by project stakeholders. A stakeholder concern rating of 1 represents a non-critical downside while a rating of 4 represents a downside of extreme importance. The external rationale factor (R) ranges from [1 – 4], with a value of 1 representing the lowest combined degree of relevance and quality in external downside rationale. The internal testing factor (T) ranges from [0 – 1] with a value of 1 representing the most significant experimental results supporting the existence of the downside and a value of 0 meaning that no significant results were found or that no evaluation was conducted for a particular downside. The mitigation factor (M) is assigned a value of either 0.5 if the downside has not been mitigated or 1.0 if the downside has been mitigated. Each of these factors in described in more detail in the following sections.

In the equation above, the internal testing factor (T) is weighted twice as heavily as the external rationale factor. This weighting reflects the idea that, if evaluations are conducted on the actual interface being developed and significant results were acquired, then those results provide must stronger evidence for the existence of a problem than the rationale provided in relation to an external system.

The stakeholder concern factor (SC) has the potential to increase significantly the combined effect of the internal and external rationale ratings if the project stakeholders are unwilling to accept the effects of the given downside. Downside mitigation reduces the probability of a risk escalating into a problem; thus, the mitigation

factor has the potential to cut the overall downside exposure value in half when a downside has been mitigated in the system design.

The resulting downside exposure value (DSE) ranges from [0 - 4]. However, DSE will only equal zero if the claim in question contains no downsides. Otherwise, DSE will be strictly positive. The range of values for each factor and the resulting downside exposure value are visualized in Figure 4.3.

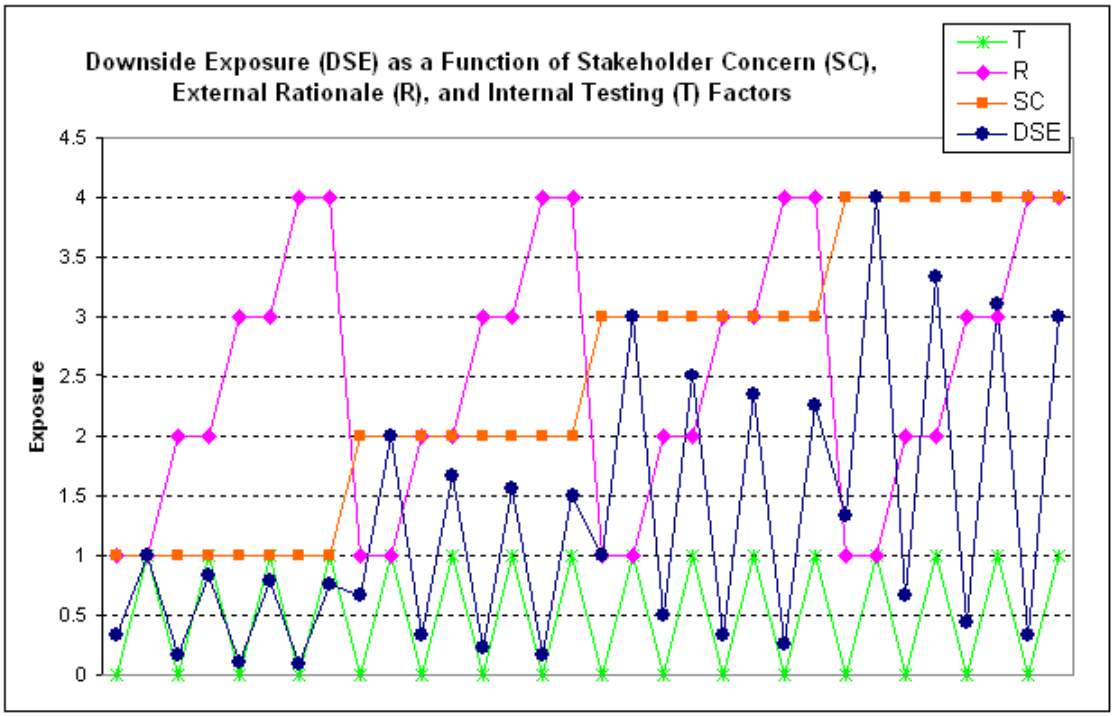

Figure 4.3: A visual depiction of Downside Exposure as a function of four key factors

4.2.2.1 Downside Mitigation Factor

A claim downside is mitigated by adding a claim to the project that has a mitigating relationship with the claim in question. For the purposes of this model, a downside is considered completely mitigated once a mitigating claim has been added to the project. Thus, the mitigation factor amounts to a binary value.

Downside mitigation reduces the probability that the downside risk will become a problem in the system. Thus, the mitigation factor (M) is factored into the downside exposure equation. To prevent multiplication by zero in the downside exposure (DSE) equation, a mitigated downside is assigned a mitigation value of 1.0 and an unmitigated downside is assigned a value of 0.5. Thus, the weight of a mitigated downside is cut in half.

In reality, downside mitigation rarely occurs in an ‘all or nothing’ fashion. Instead, multiple claims contribute to varying degrees to the mitigation of a particular downside. Consequently, the calculation of a mitigation factor is much more complex. However, for this initial risk model, the mitigation factor has been simplified to first determine the usefulness of the model as a whole and the importance of each individual factor to the estimation of risk. The results of initial evaluations of the model can lead to improvements in future versions.

4.2.2.2 Stakeholder Concern Factor

The stakeholder concern factor (SC) corresponds to the rating used in the stakeholder concern model, presented in Section 4.1. Stakeholder concern ratings measure the degree to which stakeholders are willing to accept the effect of a particular downside. A value will be assigned to each downside along a 4-point decimal scale with a minimum of 1.0 and a maximum of 4.0, based on the point indicators presented in Table 4.1 in Section 4.1.

A stakeholder concern rating of 1.0, representing a downside effect that is irrelevant to the goals of the system's end-users, has no effect on the overall exposure of that downside. In contrast, a stakeholder rating of 4.0 represents a downside effect with critical negatives consequences for the end-user. As a result, this rating has significant impact on the exposure of that downside.

4.2.2.3 External Rationale Factor

The external rationale factor (R) takes into consideration the relevance (REL) and quality (Q) of each piece of external rationale cited in the downside of a claim. If relevant rationale supports the validity of a claim downside, and if that source of rationale is of considerable quality, then system designers should be concerned that the effect of that downside might surface in their interface. The value assigned to the external rationale factor is calculated according to Equation 4.2.11.

$$R = (REL_{TOTAL})^{\frac{Q}{4}} \qquad \textbf{Equation 4.2.11}$$

Both the relevance value (REL_{TOTAL}) and the downside quality (Q) range from [1 – 4]; thus, the rationale value (R) also ranges from [1 – 4]. The relevance factor only receives its full weight in the rationale equation when the claim is of high quality, since a high quality claim is more likely to contain accurate knowledge.

The relevance factor (REL_{TOTAL}) for a source of rationale is calculated as a combination of key three factors in the design of notification systems. These factors were introduced in Chapter 3. Each source of rationale cited in a claim downside is assigned the most fitting category for each of the three factors. Additionally, the system being designed in the current project is also assigned a category for each factor.

The first factor determines the accessibility of the system. A notification system, such as a large-screen information exhibit, might be placed in a public space, for example, a hallway or the lobby of an office building. A similar display might be placed inside a particular office within that building. In this case, the display is semi-private –it is accessible to all workers within that office, but not to the general public. Finally, the display might be placed in a cubicle for the private use of one individual. This factor is assigned a value from [1 – 3] as follows:

System accessibility
- 1 Public
- 2 Semi-private
- 3 Private

The second factor considers the size of the display used by the system. A system might be deployed on a large-scale device, such as a large-screen display, on a desktop-scale device, such as a traditional desktop monitor, or on a handheld device, such as a PDA. The size of the display determines limitations of screen space, which is often a key factor in information design. Again, the factor is assigned a value from [1 – 3] according to the following scale:

Size of the system display
- 1 Large-scale
- 2 Desktop-scale
- 3 Handheld

The final factor taken into consideration is the use of the display on which the system is deployed. A system deployed on the user's primary display, such as the monitor of the user's personal computer, is forced to compete for screen space with any applications related to the user's primary task. Additionally, systems running on the primary display carry a greater risk of interrupting the user's primary task. In contrast, notification systems deployed on a secondary display do not have to compete for screen space with other applications and are more likely to remain in the user's periphery until needed. The use of display factor ranges from [0 – 1] and it assigned as follows:

Use of the display
- 1 Primary
- 2 Secondary

For each of these three factors (k), a relevance rating is calculated according to Equation 4.2.12.

$$REL_k = \frac{\min\{|\ \mathrm{Pr}oj_k - Rat_k\ |\}}{N_k} \qquad \textbf{Equation 4.2.12}$$

where N_k is the number of categories within factor k that are supported by external sources of rationale for a given downside. Thus, the numerator of this equation considers the most closely related category within factor k that is supported by external rationale, while the denominator considers how many different categories within factor k are supported by external rationale.

For example, if the system being designed in the current project is an email notification system for a handheld device and a particular downside cites external rationale pertaining to systems used on both desktop and large-screen displays, then the calculation for the size factor would be as follows:

$Proj_k = 3$

$$\min\{|\ \mathrm{Pr}oj_k - Rat_k\ |\} = \min\{|3\ 1|, |3\ 2|\}$$

$N_k = 2$

$$REL_k = \frac{1}{2}$$

Since each claim downside might have more than one source of rationale, we take the minimum difference between categories over all cited sources for a given downside and divide by the number of categories that are supported by sources of rationale. Thus, a downside with rationale supporting two of the three accessibility categories will be considered more relevant than a downside with rationale supporting only one category. If the same downside effect had been previously detected in public and semi-private displays, chances are better that it would also be detected in private displays than if it had been previously detected only in public displays.

For two of the factors, accessibility and size, the range of values for the numerator is [0 – 2], where a value of 0 means that the current project and the rationale fall into the same category and a value of 2 means that they fall into non-adjacent categories. The range of values for the denominator is [0 – 3], where a value of 0 means that no supporting rationale falls into the listed categories and a value of 3 means that supporting rationale falls into all three categories within the factor. To prevent division by zero, if $N_k = 0$, then the relevance value for that factor is set to zero. For the third factor, use of the display, vales for the numerator range from [0 – 1] and values for the denominator range from [0 – 2]. The total relevance factor accounts for all three primary factors and is calculated according to Equation 4.2.13.

$$REL_{total} = 4 - \left[\sqrt{REL_{ACC}{}^2 + REL_{SIZE}{}^2 + REL_{DISP}{}^2} \right] \qquad \textbf{Equation 4.2.13}$$

Relevance for each factor is computed as described above; thus, the accessibility and size relevance factors will range from [0 – 2] and the display relevance factor will range from [0 – 1]. The total distance factor will then range from [0 – 3], and, after the subtracting the distance factor from 4, the total relevance factor will range from [1 – 4].

The source quality factor (Q) represents the quality of the cited source of rationale. It is assigned a [1 – 4] value according to the scale outlined in the description of claim quality factors. To simplify the risk model, only downsides are considers as design risks. Thus, only downsides are considered in calculating this quality factor.

Two key factors are considered in determining quality – whether or not the work has been published and whether or not the rationale is supported by results of an empirical study. Thus, the most trustworthy, or high-quality, rationale has been published in a journal paper and has reported results from a large-scale empirical study. This quality scale is an initial, simplistic rating system that could be expanded to include a number of additional factors. A system could be developed to assign ratings to individual conferences and journals or to group venues into rating categories. However, the value of including a rationale quality rating in the risk model must first be determined. If inclusion of this factor proves significant, the method of assigning a quality rating to a source of rationale can be further explored.

4.2.2.4 Internal Testing Factor

The internal testing factor (T) takes into account the results and the confidence of any evaluations performed on the actual system being designed. When an analytic or empirical evaluation has been performed, T is calculated as a combination of two factors:

1. Effect factor (E) - the degree to which the downside affects the current design of the system
2. Confidence factor (C) - the probability that the assessment is accurate.

In cases where results of an analytic evaluation are considered, the HCI expert evaluating the system should answer the following questions about each downside:

4.2.2.4.1 Calculations for an Analytic Evaluation

The effect factor (E) for the results of an analytic evaluation will be determined by examining the interface and determining the effect of each particular downside. A value is then assigned along a 4-point decimal scale, based on the point indicators presented in Table 4.8.

Downside effect point indicators	
0.00	The effect of this downside is not an annoyance, nor is it interfering with user goal(s)
0.33	The effect of this downside is an annoyance but does not make user goal(s) more difficult to accomplish
0.67	The effect of this downside makes user goal(s) significantly more difficult to accomplish
1.00	The effect of this downside makes user goal(s) impossible to accomplish

Table 4.8: Point indicators for use in assigning downside effect ratings

The confidence factor for an analytic evaluation will be determined by the evaluator's level of experience. Based on the author experience rating used in determining claim quality, each designer will be assigned a level of experience ranging from [1 – 4], with a rating of 1.0 referring to a novice designer/evaluator and a rating of 4.0 referring to an expert designer/evaluator. Thus,

$$C = AE \qquad \textbf{Equation 4.2.14}$$

To account for the possibility that more than one individual will perform an analytic evaluation of the same interface, the internal testing factor (T) is calculated as a weighted average of all evaluated effect factors and all confidence ratings.

For example, if two individuals, with confidence ratings of 2 and 4, respectively, both evaluate an interface design and assign effect ratings of 0.00 and 0.67, then the internal testing factor will be calculates on a [0 – 1] scale as follows:

$$T = \frac{(0.33 * 2) + (0.67 * 4)}{6} = 0.56$$

A general equation for the calculation of the internal testing factor is shown in Equation 4.2.15.

$$T = \frac{\sum (E_i C_i)}{\sum C_i}$$ **Equation 4.2.15**

In cases where empirical evaluations have been conducted to test the effects of particular downsides, the experimental data should be used to calculate the effect and confidence factors.

4.2.2.4.2 Calculations for an Empirical Evaluation

The probability that a test will produce statistically significant results is termed the *power* of an experiment [19]. In other words, the power of a statistical test of a *null hypothesis* is the probability that the test will lead to the rejection of the null hypothesis, i.e., the probability that the test will conclude that the effect exists. A power analysis is often conducted prior to performing an experiment in hopes of determining *a priori* if it will produce significant results. However, the technique can also be used after the fact to indicate a degree of confidence in the results produced.

The power calculation depends on three key factors –the significance criterion, the reliability of the results of the sample, and the *effect size*, a term that measures the degree to which a particular effect exists. The significance criterion refers to the alpha value, or *alpha error*, which determines the standard for proof that a result is accurate. The alpha value is typically set to a small value, for example, alpha = 0.05, which means that the researcher is willing to accept a five percent chance that the results of the experiment will cause him (or he) to incorrectly reject the null hypothesis. In this case, the researcher assumes, based on the empirical results, that the effect being tests exists in the sample when, in fact, the null hypothesis is true.

A small alpha value results in a higher standard and, thus, a smaller probability of acquiring results that meet that standard, i.e., less statistical power. However, the complement of the power value, termed the *beta error*, measures the probability of retaining a false null hypothesis. In this case, the researcher assumes, based on the results of the experiment, hat the effect does not exist. He (or she) retains the null hypothesis when, in fact, it should be rejected. The power term increases as the beta value decreases. The researcher's goal is to balance the assigned alpha and beta values to meet the needs of the experiment.

The reliability of the sample results measures the precision with which the sample value can be used to estimate the actual population value. The exact calculation varied depending on the experiment; however, the reliability value always considers the size of the experimental sample. With all other values being equal, a larger sample size results in greater reliability in the sample results, and thus, greater statistical power.

The effect size statistic, which is of particular interest to this work, measures the degree to which a particular phenomenon, or effect, exists in the population. In other words, it measures the degree to which the null hypothesis is false. Effect size is calculated, for a two-group test, as the difference between the means for the two groups, divided by the standard deviation. Consequently, effect size measures the number of standard deviations by which the two groups differ, a value that can be used in the context of this thesis to indicate the magnitude of the impact a particular downside will have on an interface. The confidence of this measurement can be determined by calculating

a confidence interval for the effect size statistic. This calculation takes into account the size of the sample as well as a selected alpha term. Thus, the three key terms involved in power analysis are considered.

The *effect size* statistic (d), for a two-group study, is a measure of the number of standard deviations by which the two groups differ. Since effect size is dimensionless and uses a common scale to indicate the number of standard deviations by which two groups differ, the measure can be used to compare effects across multiple studies. The equation for calculating effect size differs based on the type of experiment performed. For a two-group study, effect size is calculated as shown in Equation 4.2.16.

$$d = \frac{\mu_{\exp} - \mu_{crtl}}{\sigma_{pooled}} \qquad \textbf{Equation 4.2.16}$$

where μ is the mean of the experimental and control groups, respectively. Care must be taken to perform the calculation in a manner that determines the effect size for the difference expressed in the downside.

Pooled standard deviation, σ_{pooled}, is calculated according to Equation 4.2.17.

$$\sigma_{pooled} = \sqrt{\frac{(N_{\exp} - 1)\sigma_{\exp}^2 + (N_{crtl} - 1)\sigma_{crtl}^2}{N_{\exp} + N_{crtl} - 2}} \qquad \textbf{Equation 4.2.17}$$

where N and σ are the sample size and standard deviation of the experimental and control groups, respectively.

Pooled standard deviation should only be used when it is reasonable to assume that the standard deviations of the experimental and control groups are representative of the same population and differ only as a result of sampling variation. If there is reason to believe that this assumption cannot be made, or if actual measured values for the two standard deviations differ significantly, then it is better to calculate the effect size using the measured standard deviation of the control group.

The range of values for the effect size statistic (d) typically falls between 0.0 and 1.0; however, effect size can rise above 1.0. Cohen, whose work with effect size and power analysis spans the behavioral sciences, classifies effect size according to the following scale [19]:

Small	0.2
Medium	0.5
Large	0.8 and above

To simplify the risk model, we will define the range of values for the effect size from [0.0 – 1.0]. Any d-value that falls above 1.0 will be rounded to 1.0 and any value that falls below 0.0 will be rounded to 0.0. A separate equation must be defined for a multi-group test; however, for the purposes of this initial model, calculations are restricted to a two-group test.

Effect size statistics are most useful when reported along with a confidence interval. For example, two groups may differ by an effect size of 0.5 with 95% confidence in plus or minus 0.2. In other words, we are 95% confident that the effect size falls between 0.3 and 0.7. This data represents the precision of the effect size calculation and takes into account the sample size of each group and the alpha value for the experiment.

A confidence interval for the effect size is calculated for a chosen confidence level (CL), ranging from [0 – 1] and usually falling in the .90-.99 range. For a two-group test, we find the appropriate t-value in a standard statistical table for the chosen CL and degrees of freedom (based on the sample size for the experiment). The standard deviation of the effect size is then calculated as shown in Equation 4.2.18.

$$\sigma_{[d]} = \sqrt{\frac{(N_{\exp} + N_{crtl})}{(N_{\exp} * N_{crtl})} + \frac{d^2}{2(N_{\exp} + N_{crtl})}} \qquad \textbf{Equation 4.2.18}$$

where d is the effect size and N is the sample size for the experimental and control groups, respectively. The confidence interval is then calculated as shown in Equation 4.2.19.

$$d \pm t\sigma_{[d]} \qquad \textbf{Equation 4.2.19}$$

Again, a separate equation must be defined for a multi-group test.

Given the confidence interval, the internal testing factor, T, is calculated according to Equation 4.2.20.

$$T = \begin{cases} d(CL). & if \ \frac{d}{t\sigma_{[d]}} > 0 \\ 0 & if \ \frac{d}{t\sigma_{[d]}} \le 0 \end{cases} \qquad \textbf{Equation 4.2.20}$$

where d is the effect size, CL is the chosen confidence level, t is the t-value for confidence level CL, and $\sigma_{[d]}$ is the standard deviation of d.

If the confidence interval is strictly positive, then the results of the experiment show a measurable difference between the two groups to the level of confidence chosen. In this case, T is defined as d*CL, i.e., the impact of the downside multiplied by the probability that the results are accurate. However, if zero is included in the confidence interval, then no significant results can be shown at the chosen confidence level. In this case, T = 0.

After the completion of one evaluation, be it analytic or empirical, some form of redesign will presumably be conducted before performing another evaluation. Thus, only one set of internal testing factors (effect and confidence) will be included in the risk model at any given time. The results of the most recent evaluation are included in the model.

4.3. Risk Model Implementation within LINK-UP

As discussed in Chapter 2, the LINK-UP system guides designers through a claims-centric, scenario-based usability engineering process. The system currently includes two key modules, which walk designers through the steps of requirements analysis and the creation of a system image. This section explains the integration of a Risk Analysis module within LINK-UP, allowing for the collection of risk-specific information, the dynamic calculation of claim risk exposure values, and the prioritization of project claims.

The Risk Analysis module is designed for use in conjunction with the Requirements Analysis and System Image modules in LINK-UP. Data is collected by the Risk Analysis module throughout the design process, beginning with high-level system design characteristics early in design and continuing with the results of participatory design sessions and design evaluations.

Several factors have been added to the structure of a claim for use in the claim quality and risk calculations described in previous sections. For each upside and downside added to a claim, the claim author is asked to choose the publication venue and type of results that best matches each external source of rationale listed in support of the claim. The author is also asked to supply the most appropriate relevance factors (accessibility, size, and use of display) for each source of rationale so that these values can be compared to the overall system characteristics. All of these factors are collected during claim creation and stored with the claim in the knowledge repository for reuse in subsequent projects. A screenshot of the claim creation module within LINK-UP is presented in Figure 4.4.

The Risk Analysis module also allows designers to input stakeholder concern ratings for each downside of each claim in their project set along with the results of one or more analytic evaluations of the interface. The module follows the conventions of other aspects of the system, breaking the claim set up by stage of action and allowing the designer to work with one claim at a time. Progress indicators aid designers in determining how far they have progressed through the claim set, and they alert designers when data has been input for all claims in the set. A screenshot of the stakeholder concern rating input section of the Risk Analysis module is presented in Figure 4.5. The analytic input section of the module is implemented in a similar manner.

The most important aspect of the Risk Analysis module is the prioritized claim list made available to the designer throughout the project. With this list, designers can identify the most critical design risks to be addressed during the current design iteration. Additionally, designers can monitor lower-priority risks and their evolution over time. The list updates dynamically based on changes to the project claim set. A screenshot of the claim priority list output from LINK-UP is depicted in Figure 4.6.

Both the stakeholder concern model and the comprehensive risk model have been implemented within LINK-UP. Although the appearance of the Risk Analysis module remains the same, the backend calculations used in prioritizing project claims can be altered by a LINK-UP administrator. As a result, the stakeholder concern model and the comprehensive risk model can be used in calculating risk for different project teams without those teams being aware of the specific model being used. This feature was added to facilitate the user evaluation, which is presented in detail in Chapter 5.

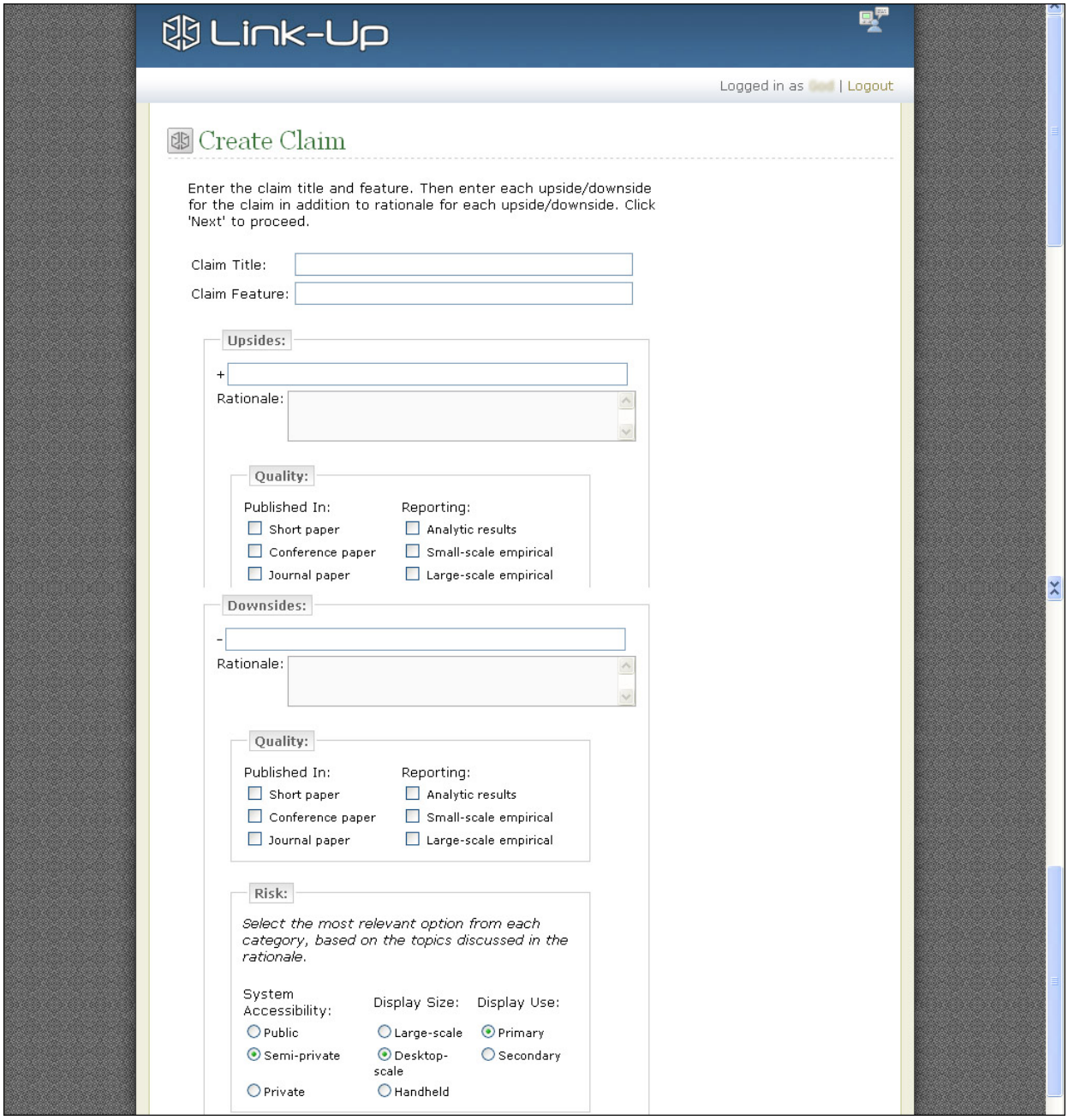

Figure 4.4: Screenshot of the claim creation module within the LINK-UP Risk Module

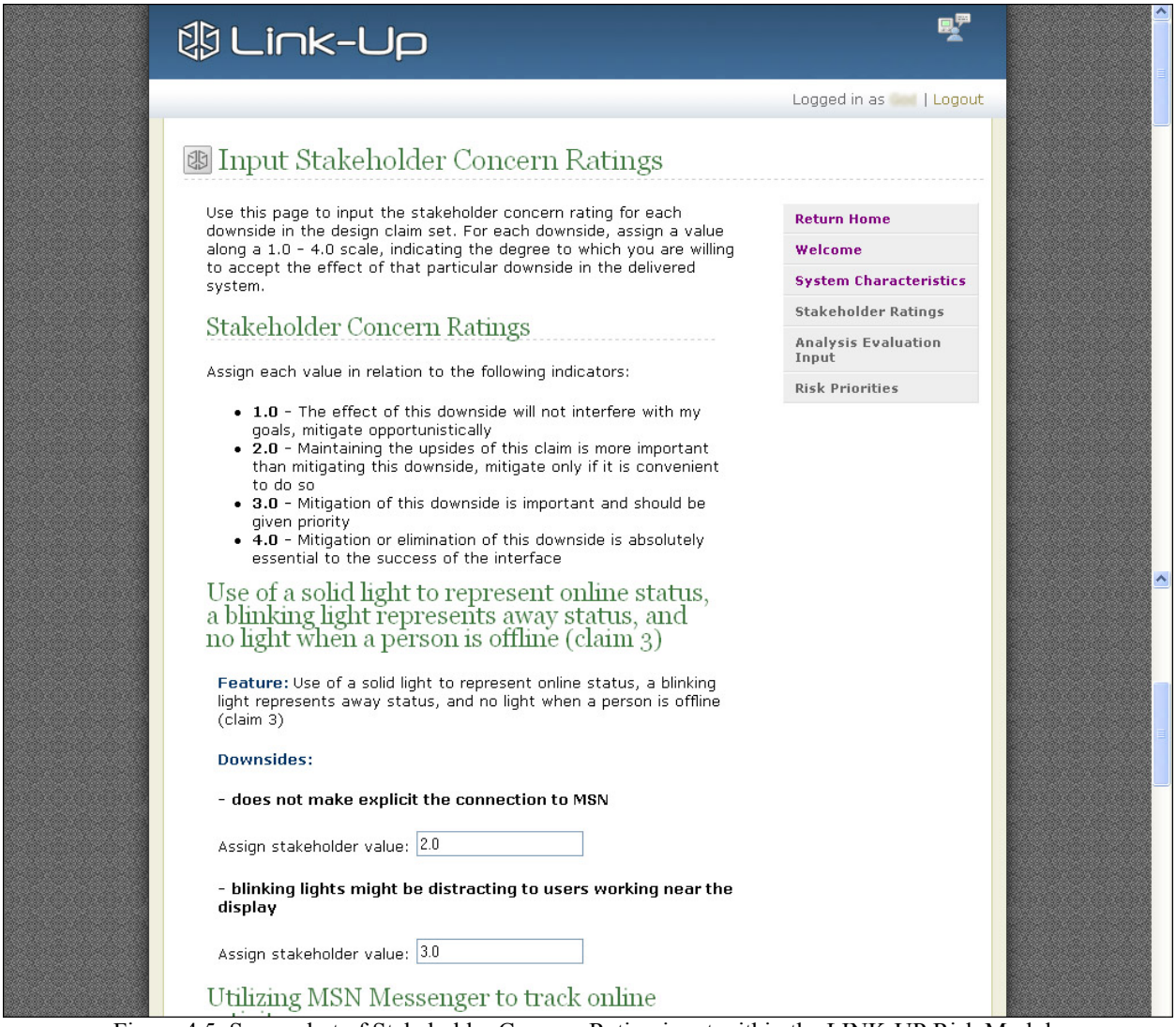

Figure 4.5: Screenshot of Stakeholder Concern Rating input within the LINK-UP Risk Module

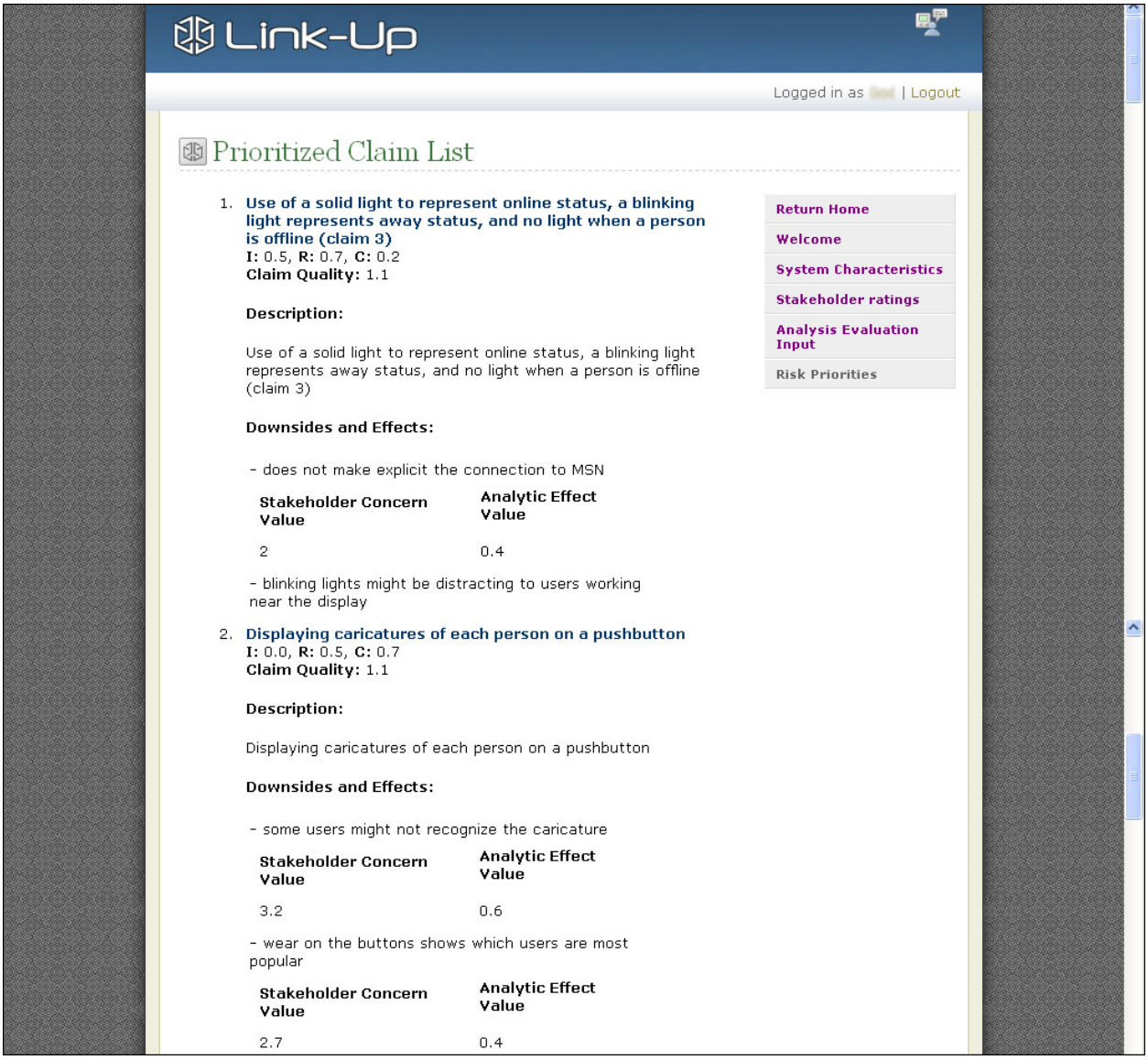

Figure 4.6: Screenshot of the claim priority list within the LINK-UP Risk Module

4.4. Flexibility and Limitations of Integration within SBD

The key goals of this work are to further the science of design while improving the standards by which software projects are managed. To be useful to designers, a risk model must provide accurate estimates of design risk while adding minimal overhead to a project. Consequently, design knowledge for use in the model must be collected, whenever possible, as the byproduct of other aspects of the design process. This initial version of the comprehensive risk model attempts to integrate the collection and calculation of risk exposure values directly into accepted phases of the SBD process. However, the underlying goal in developing this model is to determine the usefulness of integrating risk management into HCI design; thus, this initial model represents a proof of concept as opposed to a finalized, usable risk model. This section discusses not only the flexibility shown by the initial model, but also several limitations that must be addressed as the model is refined through future versions.

Ideally, use of a risk model should not interfere with the design process, and the comprehensive model minimizes overhead whenever possible. Each piece of information collected for use in the model also plays a role in another aspect of the design process. For example, data collected during claim creation contributes to the claim quality and external rationale calculations. This information is collected from the claim author and can be reused from project to project; thus, no overhead is added to the current design project. Stakeholder concern ratings result from a participatory design session, an analytic and empirical effect ratings result from respective evaluations. These activities are already accepted aspects of the design process, and collecting specific information serves to add structure to participatory design and evaluations. Additionally, collection of information at various stages of the design process from various designers and other project stakeholders reduces the amount of effort required from any one member of a project team. In this way, risk management becomes a group activity as opposed to the responsibility of a project manager. The process of gathering risk-related factors is divided between the designer base and the user base for the system, with emphasis somewhat skewed toward the users. Stakeholder concern ratings and internal evaluation factors, which account for a significant portion of the total risk equation, are both provided by end-users, or, at the very least, end-users should be involved in the process. This emphasis on user satisfaction represents the overall goals of HCI and scenario-based design.

Furthermore, risk exposure can be calculated at any point during the design process using whatever information is currently available. Although the accuracy of the risk estimates improves as more information becomes available, the ability to identify and prioritize risks early in the design process is invaluable. Ideally, risk exposure values will be calculated based on a base exposure rating, the quality and relevance of external sources of rationale, and the results of stakeholder negotiations and internal evaluations of the system being designed. If the results of an empirical evaluation are not available, then analytic results can be used. If no evaluations have taken place, then risk exposure can be based on external rationale, claim quality and IRC difference values. If no rationale is available, then this factor can also be left out of the equation. Moreover, stakeholder concern ratings can result from varying degrees of stakeholder involvement. Ideally, these ratings will be collected from a number of stakeholders during a participatory design session. If project constraints do not allow for such a session, then one project stakeholder, the design teams as a whole, or even a single member of the design team, can assign concern ratings on the stakeholders' behalf. Although the usefulness of the concern ratings will increase with the involvement of more stakeholders, it is possible to acquire a meaningful result with little or no stakeholder involvement.

Implementation of the risk model within LINK-UP allows designers to frequently view an up-to-date priority list of their project claims. This list is updated dynamically as the design evolves, reflecting changes, such as the addition of new project claims, changes in claim quality over time, and mitigation of particular downside risks. Consequently, designers can continue to utilize the list throughout multiple design iterations.

Scalability at a high level is improved by the decision to prioritize claims instead of individual downsides risks. Nevertheless, as design projects increase in size and complexity, the number of claims needed to document a robust system design increases as well. Fortunately, the goal of the risk model is to draw the most critical design risks to the top of the priority list. As a result, project teams can allocate resources to the largest number of risks that project constraints will allow. Additionally, portions of the priority list, in the form of individual downsides, claims, or

clusters of related claims, can be allocated to individuals or subsets of the project team, based on the size of the team and other project constraints. At a lower level, several of the risk calculations that rely on hard numbers are not as scalable. Since this initial set of equations is more of a proof of concept than a finalized model, these issues must be addressed in future versions.

Although the model is flexible in many respects, this initial version also has limitations. For example, although risk exposure can be calculated using whatever information is available, it is difficult to compare and prioritize claims with risk exposure values based on different subsets of information. A claim that has not undergone any analytic or empirical evaluations will have an internal testing factor of zero ($T = 0$). The lack of internal testing will lower the overall risk exposure value for that claim; however, the claim might still be a critical risk. In an attempt to curb this behavior, the stakeholder concern rating, which can be applied to each downside at any point during the design process, has been given significant pull in the equation for downside exposure. However, it is not clear exactly how claims with different pieces of available information should interact within the risk model. A second limitation involves the empirical testing calculations. Currently, equations have been defined only for a two-group t-test. Additional equations are needed for different types of evaluations, which might complicate use of the model.

Although these limitations must be addressed in future versions of the risk model, this initial version represents an adequate proof of concept for use in evaluating the usefulness of risk management in the context of HCI design and also in determining the most critical factors for use in estimating design risk. The experimental design and results of an initial user evaluation are presented in detail in the next chapter.

Chapter 5 : Risk Model Evaluation

Integration of a risk-driven, claims-centric model within the SBD process is meant to improve the structure of the design process. Additionally, use of the model should improve management in design projects by automating risk management and facilitating team coordination with minimal added overhead. Chapter 3 outlines the benefits of developing a management model for product-related design risks and discussed the key factors to consider in mapping HCI concepts to existing risk management paradigms. The chapter hypothesizes that the integration of risk management within the SBD process will help design teams to identify and focus on critical issues for the redesign of a system interface. Additionally, the chapter hypothesizes that scenarios and claims, the key components of SBD, will aid in the storage and reuse of risk-related knowledge, support integration of the risk model within the SBD process, and facilitate discussion among designers and project stakeholders. These and other issues, such as the benefit of prioritizing claims as opposed to individual downsides and the degree to which the risk model supports task allocation, will be the focus of the user evaluation discussed in this chapter.

The overall goals for this initial evaluation of the risk model are first to determine the usefulness of risk management in an HCI design process and then to determine the accuracy of the risk estimates calculated by both the comprehensive and stakeholder concern models. More specifically, student design teams will use and assess the risk model throughout several phases of a semester-long design project. Given a prioritized list of their project claims, based on the results of one model or the other, these teams will be asked to reprioritize the list as they see fit. The degree to which teams update the priorities of their project claims will then be used to measure the extent to which designers agree with the results of each risk model. The design teams will also be asked to report on the specific project downsides that they mitigated during a redesign phase of the project. A correlation between the priority of claims from which downsides were chosen for mitigation and the grade received for the redesign phase of the project will be used to determine the extent to which use of the risk model improves design quality. Throughout several project phases, the designers will also be asked a series of subjective questions to determine the benefit of prioritizing claims versus downsides, the extent to which SBD contributes to risk management, and the degree to which use of the risk model improves team coordination by facilitating task allocation.

Section 5.1 begins by outlining a set of key research questions to be answered through this evaluation. These questions are answered through a case study of an existing system design as well as a more in-depth user evaluation. The results of these evaluations are presented in detail throughout the remainder of the chapter.

5.1 Research Questions

The goal for this evaluation is to answer six key questions. These questions depict the overall goals of this thesis by establishing the benefit of applying risk management techniques within the context of HCI design and by determining the most critical factors to consider in an estimate of design risk. Additionally, these questions draw

together several reoccurring themes, such as SBD, design risk, and team coordination. The six key research questions are presented below.

First, are claims better suited for prioritization than individual downsides? Section 3.3.1 discusses the decision to prioritize project claims as opposed to individual downside risks both to preserve the fundamental unit of design knowledge in the form of a claim and to increase scalability of the model at a high level. This question seeks to determine to what degree designers agree with that decision.

Second, the evaluation should determine to what extent a risk-driven model will contribute to the redesign phase of an HCI design project. Section 3.3.2 discusses the ways in which risk management can support SBD by aiding in the improvement of a project claim set. Does the model aid designers in deciding which downside risks are most important to mitigate during the redesign of their interface?

Third, how does scenario based design contribute to risk management? Section 3.3.3 discusses the integration of risk management tasks within several SBD activities. The section also discusses the structure of claims and the usability of scenarios in organizing and communicating design knowledge. What characteristics of scenarios and claims facilitate the organization and communication of design risks?

Fourth, how can the risk model improve team coordination? Section 3.3.4 discusses the contribution of risk management in supporting and monitoring task allocation. The mitigation of individual downsides can be viewed as project tasks. Does use of a risk model support this view and aid designers in determining and allocating key project tasks?

Fifth, the evaluation should determine to what extent designers agree with the results of the risk model. Section 3.3.5 explains the overall process by which project claims will be prioritized. The details of risk exposure estimation, based on two separate risk models, are then presented in Chapter 4. This question seeks to determine the accuracy of the estimation process. Does the prioritized claim list output by the risk model place the most critical design risks at the top of the list?

Finally, the evaluation should determine what factors are most important to consider in an estimation of design risk. Section 3.3.5 also discusses the key factors considered in the comprehensive risk model and rationalizes why each factor should be included in the risk estimation. Does a comprehensive model, including all available information, result in a more accurate risk estimate than a simpler model that only considers one critical risk factor? These six research questions are summarized in Table 5.1.

Research Questions	
1	Are claims better suited for prioritization than individual downsides?
2	How well does each model contribute to the redesign phase of a project?
3	What do scenarios and claims contribute to the management of design risks?
4	Does use of a risk model aid in determining and allocating key project tasks?
5	To what extent do designers agree with the prioritizations of their project claims?
6	What factors are most important to consider in an estimation of design risk?

Table 5.1: Key research questions to be answered through evaluation of the risk model

The next section presents a case study of an existing notification system that was designed using the SBD process. The claim set for this project was prioritized using both the comprehensive and stakeholder concern models, and the lead designer on the project was asked to assess the results and address the research questions outlined above. Section 5.3 then presents the experimental design for a more in-depth user evaluation involving approximately 20 undergraduate design teams. These teams were asked to use and assess the risk model during several phases of a semester-long project in which they were using SBD in the design of simple notification interfaces. Section 5.4 presents and discusses the results of the user evaluation with respect to each of the six research questions. Finally, Section 5.5 summarizes the evaluation results.

5.2 Online Enlightenment Case Study

Before the risk model was evaluated on a significant set of design projects, it was used to prioritize the claim set of an existing system. The Online Enlightenment project [28] created an off-the-desktop notification system to alert users of an HCI lab as to which of their lab partners are currently available through an online instant messaging system. Since instant messaging has become a staple in the professional world for use in coordinating project tasks, meetings, and deadlines, regular lab users, as well as visitors, can benefit from the ability to see who is available without logging into the messaging system on an individual computer.

The design of OE followed a claims-centric, scenario-based approach. As an existing notification interface with a documented set of scenarios and claims, OE presented the ideal opportunity for a case study. An initial version of the system had been implemented and deployed, and, after several months of use, the design team was contemplating improvements. Consequently, an assessment of the system using both the comprehensive and stakeholder concern risk models provided a valuable test bed for the risk model as well as a beneficial assessment for the OE design team.

OE is a physical device that presents online presence information in a semi-public space. The device uses a map metaphor to represent the layout of a lab, showing online instant messenger status for members of the community. Users of the device can combine information from the device with information from their lab physical environment to identify unfamiliar lab members, determine human-to-human interaction strategies, and plan meetings. Development of the OE interface was inspired by lab users' desire to locate other members of the lab, as well as the lab director's desire to keep track of his research students. Additionally, the system was developed as a physical example of a notification system, allowing lab visitors to not only locate members of the lab, but also understand the research taking place within the lab. A problem scenario and sample problem claim, adapted from [28], are presented in the next section.

5.2.1 Problem Description and Root Concept

The Lab Users Problem Scenario, depicted below, illustrates the need for a system such as OE.

> *Jason is working at this computer in the HCI lab. He asks Ali a question and is told that Christa would have the necessary information. Jason checks his MSN buddy list to find that Christa is not currently online. He configures MSN to sound an audible alert when Christa signs into the system, then minimizes his buddy list window and continues working. Unfortunately, if the tone sounds while Jason is out of the room, he will not*

hear it. When he returns to the lab, the only way to find out if Christa has signed into MSN is to restore his buddy list window and check her status.

Jason continues working at his computer, leaving the lab several times to run errands in other parts of the building. He gets his lunch and then returns to his desk to continue working. After a while, Jason checks his MSN buddy list to find that Christa has been signed on for almost an hour. The tone must have sounded while he was out of the room.

A sample problem claim for the OE system is depicted in Figure 5.1. Additional problem claims for the system are presented in Appendix A.

Finding all the lab users using MSN	**IRC:** 1.0, 0.8, 0.1
+	easy to get all the information you need at once
+	intuitive to respond if needed
-	requires access to a computer
-	becomes the user's primary task
-	must continually check the list to monitor status of lab users
-	no history of information regarding the state of the user over a period of time
-	no information of how long the particular user has been in that state for

Figure 5.1: An example OE problem claim

A centrally located display providing current and historical information about lab users could improve the productivity of the lab by supporting lab users in locating and communicating with one another. The system must promote easy reaction to the information made available as well as minimal diversion from users' primary tasks. Minimizing attention diversion is important because many regular lab users would experience a decrease in productivity if the notification system constantly interrupted their normal tasks. By making status information easy to understand, the amount of time spent interfacing with the system would be minimal, allowing users to perform their normal tasks. OE is being developed to promote communication within the lab, since the research of many lab users is inter-connected and dependent upon frequent collaboration. By promoting communication, the productivity of the lab would be subsequently increased.

The root concept for the system is summarized in Figure 5.2, which depicts the primary and secondary goals of the system as well as the target IRC rating for the system. The target IRC rating for system is (0.0, 0.6, 1.0), indicating a low level of interruption, a moderate level of reaction, and a high level of comprehension.

Primary Goals:	
Provide easy reaction to information regarding the availability of lab users	
Minimize interruption to users' primary tasks	
Secondary Goals:	
Tie the system to an existing instant messaging system	
Attract and impress lab visitors	
Target IRC:	(0.0, 0.6, 1.0)

Figure 5.2: Summary of OE system goals

The problem scenario and root concept for this system could be approached from various angles, each of which presents a number of design risks. Through using the risk model, the OE design team can assess the risks of their design solutions and determine which risks should be addressed during the next design iteration. The next section outlines the assignment of a number of factors that will be used in estimating risk exposure values for the OE claim set.

5.2.2 System Characteristics

The first step in utilizing the risk model is to assign a number of values, including the design model IRC rating and system design characteristics, for the OE system. The system will be deployed on a desktop-scale, secondary display in a semi-public space. These values are summarized in Figure 5.3.

Design size:	(2) Desktop-scale
Design accessibility:	(2) Semi-private
Display use:	(2) Secondary

Figure 5.3: OE system characteristics

A subset of the OE claim set was chosen for use in this study. For each of the eight chosen claims, all values needed for the risk calculations were assigned by users and designers of the system.

Several values involved in the claim quality calculations remain the same for all eight claims. These values include the experience of the claim author, the type of artifact associated with the claim, the degree of reuse of the claim, and the average user rating for the claim. The assigned values are summarized in Figure 5.4.

Author experience:	(2) Novice designers created these claims
Artifact quality:	(2) As OE was being designed, simple screenshot prototypes existed
Degree of reuse:	(1) All new claims, nothing has been reused
Average user rating:	(1) None of the claims have been rated

Figure 5.4: Claim quality characteristics common to all OE design claims

The design proceeded through activity, information, and interaction design, with the goals of mitigating the downsides listed in the problems claims while maintaining or strengthening the upsides. A sample design claim is presented in Figure 5.5. Along with the claim feature, tradeoffs, and IRC rating, each design claim lists the overall claim quality rating, the venue and results factors for each tradeoff that contribute to the quality of rationale, the system relevance characteristics for each source of rationale attached to a downside, the stakeholder concern rating and analytic effect value for each downside, and the risk exposure value for each downside. All eight design claims used in this case study are presented in Appendix A.

Each claim lists the claim feature, claim IRC value, and claim quality rating. For each tradeoff associated with the claim, venue and results value depict the type of venue in which the rationale supporting that tradeoff is published and the type of results reported in the publication. These values contribute to the quality rating of the

claim. The stakeholder concern (stake) and downside effect (effect) ratings are listed for each downside, as well as the accessibility, size, and use characteristics assigned to each source of rationale. Mitigation of each downside (mit) is also shown, with a value of 1.0 depicting no mitigation and a value of 0.5 depicting full mitigation of the downside. Finally, the risk exposure value for each downside (DSE) is shown, with the highest DSE within each claim in bold. This maximum DSE value then contributes to the overall risk exposure value for each claim.

Utilizing MSN Messenger to track online activity (claim 1)							IRC: 0.0, 0.8, 0.1			
Claim Quality: 1.5		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	***DSE***
+	easy to collect necessary information	3	2	x	x	x	x	x	x	x
+	provides information about an individual, not a computer	3	2	x	x	x	x	x	x	x
-	requires access to a computer	3	2	2.0	0.1	2	2	2	1.0	0.41
-	cannot distinguish users in the lab from remote users	3	2	2.5	0.4	2	2	2	1.0	1.02
-	users might forget to log into MSN	3	2	1.5	0.2	1	2	1	1.0	0.48
-	no history of information about the state of the user over time	3	2	3.0	0.6	2	2	2	1.0	**1.62**

Figure 5.5: Example OE design claim used in prioritization

5.2.3 Prioritized Claim Lists

Once the necessary values were assigned to each of the eight design claims, the claims were prioritized according to two models. The first is a comprehensive model including all factors presented in Chapter 4. The second model is a simplified version, which considers only the stakeholder concern ratings for each downside. The stakeholder concern ratings are averaged across all downsides within a claim, and that average represents the overall risk exposure value for that claim.

The priority list resulting from the comprehensive model is depicted in Table 5.2. The overall risk exposure value for each claim, calculated on a scale from 1.0 to 4.0, is shown along with the claim feature and claim priority.

Results based on comprehensive risk model		
Priority	Risk Exposure	Claim feature
1	2.86	Use of a solid light to represent online status, a blinking light represents away status, and no light (claim 3)
2	2.41	Utilizing MSN Messenger to track online activity (claim 1)
3	2.27	Displaying caricatures of each person on a pushbutton (claim 5)
4	1.97	Centrally located physical device (claim 2)
5	1.91	Use of pushbuttons for each person to change LCD information (claim 4)
6	1.88	LCD displaying textual information (claim 6)
7	1.38	Display depicting a scaled map representation of the lab space (claim 7)
8	0.98	Use of a wall-mounted display (claim 8)

Table 5.2: Priority list of OE design claims calculated using the comprehensive risk model

The priority list resulting from the simplified, stakeholder concern model is presented in Table 5.3. Again, the overall risk exposure value for each claim is shown along with the claim feature and claim priority.

Results based on stakeholder concern model		
Priority	Risk Exposure	Claim feature
1	3.0	LCD displaying textual information (claim 6)
2	2.75	Display depicting a scaled map representation of the lab space (claim 7)
3	2.73	Displaying caricatures of each person on a pushbutton (claim 5)
4	2.70	Use of pushbuttons for each person to change LCD information (claim 4)
5	2.50	Use of a solid light to represent online status, a blinking light represents away status, and no light when a person is offline (claim 3)
6	2.275	Centrally located physical device (claim 2)
7	2.25	Utilizing MSN Messenger to track online activity (claim 1)
8	1.50	Use of a wall-mounted display (claim 8)

Table 5.3: Priority list of OE design claims calculated using the stakeholder concern model

5.2.4 Designer Feedback

The lead designer on the OE project, Goldie Terrell, participated in an informal evaluation of the risk model. She examined the priority lists resulting from both risk models and answered a number of questions concerning both the accuracy and usefulness of risk management in HCI design. Terrell was presented with the OE case study provided in Appendix A, including an explanation of the goals of the risk model and the key factors considered the estimation of design risk. Although she viewed the priority lists resulting from both models, she was not told which list resulted from which model.

Terrell was first asked to compare the two priority lists and determine which list seemed more accurate to her in terms of drawing the most critical design issues to the top. In doing so, she chose to create her own priority list and then compare it to the two existing lists. Terrell's priority list is presented in Table 5.4.

OE designer's claim prioritization	
Priority	Claim feature
1	Utilizing MSN Messenger to track online activity (claim 1)
2	Use of a solid light to represent online status, a blinking light represents away status, and no light when a person is offline (claim 3)
3	LCD displaying textual information (claim 6)
4	Centrally located physical device (claim 2)
5	Display depicting a scaled map representation of the lab space (claim 7)
6	Displaying caricatures of each person on a pushbutton (claim 5)
7	Use of pushbuttons for each person to change LCD information (claim 4)
8	Use of a wall-mounted display (claim 8)

Table 5.4: Terrell's prioritized list of the eight OE design claims

Terrell's list most closely resembles the priority list resulting from the comprehensive risk model. She chose to flip the positions of the first two claims, but maintained their position at the top of the list. Likewise, she maintained the position of the last claim in the list. The positions of claims 3 and 6 were flipped in Terrell's list, as were claims 5 and 7. Claim 4 remained in its original position. This reprioritization maintains the extreme rankings in the list, while the intermediate rankings shifted. Since these intermediate claims have more closely related risk exposure values than do the extremely ranked claims, they are more likely to be shifted in priority due to the differing opinions of various designers.

Terrell was then asked to consider the highest priority downside in each of the eight OE claims and comment on her agreement with that downside being the most critical risk within the given claim. In seven out of eight cases, she agreed that the downside calculated by the comprehensive risk model as the highest priority downside was in fact the most critical risk. In one case, she felt that another downside in the claim was more critical, which was likely the result of differing opinions in terms of the stakeholder concern ratings assigned to those downsides.

After examining the various downsides contained within the eight OE claims, Terrell listed what she felt were the four most critical downsides throughout the claim set as a whole. These downsides are listed in priority order in Table 5.5. Each of her top four downsides is a tradeoff listed in each of the top four claims as ranked by the comprehensive model. The first and second priority downsides –the blinking lights on the display and the lack of history information provided by MSN, are so critical that they have already been mitigated during a redesign of the system.

Next, Terrell was asked to rank the five key factors considered in the comprehensive risk model, namely, stakeholder concern ratings, internal evaluation results, claim quality, difference in IRC values, and quality and relevance of external rationale. Her rankings are presented in Table 5.6.

Terrell commented that the top two factors in her ranked list –internal evaluation results and stakeholder concern ratings, are the most critical because they incorporate the user's perspective as opposed to the designer's perspective. Furthermore, the results of an internal evaluation provide information about an existing system in use, whereas the stakeholder concern ratings provide the user's thoughts before a system is actually developed for use.

OE designer's top priority downsides		
Priority	**Downside**	**Claim**
1	Blinking lights might be distracting to users working near the display	Claim 3
2	No history of information about the state of the user over time	Claim 1
3	Use of the display in a semi-public area raises privacy issues	Claim 2
4	Not obvious that the caricature is a pushbutton	Claim 5

Table 5.5: Terrell's ranking of four highest priority OE downsides

Consequently, Terrell ranked internal evaluation results above stakeholder concern ratings in terms of importance in estimating risk. She ranked external rationale third because it provides reasons for why a particular downside might be a problem in a design and allows the designer to consider the effect of a particular claim on the system as a whole. She ranked IRC difference fourth because, although she was unsure of its impact on risk exposure, IRC values are still concerned with design within the context of a particular system.

OE designer's ranking of risk factors	
1	Internal evaluation results
2	Stakeholder concern ratings
3	External rationale
4	IRC difference
5	Claim quality

Table 5.6: Terrell's ranking of the five key risk factors considered in the comprehensive model

Claim quality was ranked fifth because, although it calculates the probability of having a high quality claim, Terrell felt that it does not provide an absolute claim quality value. In particular, Terrell mentioned that the claim quality calculation does not take into account the probability that the claim author has considered all possible design tradeoffs. Although this is a difficult value to measure, it is important in determining the quality of a claim. Additionally, the risk model as a whole does not account for dependability among claims and the effect that it might have on individual downside risks. Dependability among claims makes risk prioritization a more difficult task, reducing the accuracy of the results based on the existing comprehensive model.

When asked whether she, as a designer, would prefer a prioritized list of claims or individual downsides, Terrell chose downsides. In support of her decision, she cited the example of two critical downsides within the same claim. When given a prioritized list of claims, Terrell naturally examined the highest priority downside in each claim when considering which downsides to mitigate. If more than one downside in the same claim is worthy of mitigation, she felt that those risks should be listed separately to draw attention to their high priority. Additionally, she mentioned that it felt more natural to think directly in terms of downside risks and the particular problems to be solved.

Although she preferred a priority list of downsides, Terrell stressed the importance of linking the downside to the claim in which it appears. The context of the design feature, along with the additional design tradeoffs within the claim, are critical considerations when determining whether a downside should be mitigated and how that mitigation might be accomplished without negatively impacting the design.

Finally, Terrell was asked to comment on how she might use the prioritized claim list during the redesign phase of a project. At a high level, she discussed working through the prioritized claim list from top to bottom, examining the downsides of each claim and considering mitigation strategies. In doing so, she would consider the stakeholder concern ratings and internal evaluation effects for each downside, thinking about how the downside could be mitigated and how costly the mitigation would be. The benefit of having an ordered list is that she can easily

determine which problems are worth solving first, solve as many problems as project constraints will allow, and be confident that she has solved the most critical problems.

5.3 Experimental Design

Following completion of the OE case study, the risk models were evaluated by a number of undergraduate design teams. The overall goals of the evaluation were to determine the usefulness of risk management within the context of HCI design and to assess the accuracy of the initial, proof of concept calculations. Regardless of the accuracy with which the two initial models estimate risk, the first step in the evaluation process is to establish the benefit of integrating risk management within the SBD process. The factors determined to be critical to an accurate estimation of design risk can then be reassessed in a second version of the risk model.

Evaluation of the risk model was incorporated into the semester design project for an undergraduate, introductory HCI course. Fifteen teams of approximately four students each were given the task of designing a navigational notification system. The teams were divided into two sections of the same course. All teams used LINK-UP throughout the semester to guide their design process and manage the knowledge resulting from their design effort. The remainder of this section describes the procedures through which this evaluation was conducted along with the types of data that were collected to answer the research questions outlined in Section 5.1.

5.3.1 Preparatory Assignments

The student design teams were asked to collect specific information throughout several phases of their design project and input that information into the LINK-UP Risk Analysis module. This information was used in calculating a prioritized claim list for each project team, which they then used to guide the redesign phase of the project.

During claim creation, students were asked to provide information about both the quality and relevance of external sources of rationale. This information was input into LINK-UP and stored within the structure of a claim for reuse. Following the initial design phase of the project, teams were asked to act on behalf of their target end-user and other project stakeholders to assign stakeholder concern ratings to each downside in their project set. During an analytic evaluation exercise, in which one project team evaluated the interface of another team, students were asked to assign a downside effect rating to each downside in the project set. All of this information was input into LINK-UP and used in the risk calculations. After recording this information, teams were asked a follow-up question to determine whether claims and scenarios contribute positively to group discussions about design.

After inputting this information into LINK-UP, students were asked to assess the prioritized list of their project claims produced by LINK-UP. The same two risk models used in the OE case study were also evaluated at this stage of the experiment. The first is the comprehensive model including all factors presented in Chapter 4 (note that the results of an analytic evaluation were used instead of empirical results, and for this initial redesign the mitigation factor of each downside defaulted to a value of 1.0). The second model is the simplified version, which considers only the stakeholder concern ratings for each downside.

The nine design teams in one section of the course were presented with a prioritized claim list based on the comprehensive risk model, while the thirteen design teams in the other section were given results based on the stakeholder concern model. The teams were not told which model was used to produce their priority lists.

5.3.2 Evaluation Procedure

Each team was asked to complete the chart in Figure 5.6 for each claim in their project set. The first task was to re-prioritize their claim set as they saw fit, providing rationale for their decisions to increase, decrease, or maintain the priority of each claim. Possible characteristics supporting priority refer to the key components of the risk model and are outlined in Table 5.7.

Characteristics Supporting Priority	
1	**Claim quality**: the quality of a particular claim as a whole, which takes into account the experience of the claim author, the quality of external sources of rationale, the quality of the any artifact associated with the claim, the degree to which the claim has been reused, and an average user rating for the claim
2	**IRC difference**: the degree of difference between the claim IRC rating and the design model IRC rating
3	**Stakeholder concern ratings**: the degree to which project stakeholders are willing to accept the effect of a particular downside in the design of the system
4	**External rationale**: the degree to which external sources of rationale supporting a particular downside present evidence of its effect in similar systems
5	**Internal evaluation results**: the degree to which the results of an evaluation of the interface show evidence that a particular downside is effecting the interface
6	**Other**

Table 5.7: Possible claim characteristics supporting update or maintenance of risk priority

The "Explanation" column of the chart asks students to elaborate on the reasoning behind their decision to update the priority of each claim. For example, if the *Claim quality* characteristic affects the team's decision to raise the priority of a particular claim, they might elaborate by stating that:

> "The quality of this claim is very low and I don't see where this characteristic factors into the prioritization"

If the *Stakeholder concern rating* characteristic affects the team's decision to lower the priority of a claim, they might elaborate by stating that:

> "Project stakeholders are not very concerned about any of the downsides in this claim, and the claim seems to be too high on the list"

And if the *Internal evaluation results* characteristic affects the team's decision not to change the priority of a claim, they might elaborate by stating that:

> "Results of the analytic evaluation show that none of the downsides in this claim are a problem, and the priority seems to accurately reflect those results"

Students were asked to provide characteristics and an explanation for each claim in their project set regardless of whether the priority of the claim was altered or maintained.

After assessing the priority of each claim, the students were asked to choose a subset of project downsides that they felt were most important to mitigate during the redesign phase of the project. Given time constraints, students were allotted only one week for redesign; thus, they were encouraged to choose a small subset of downsides that were critical risks but also that were feasible for mitigation in that time frame. Finally, students were asked to answer four follow-up questions. The first two questions assess the benefit of prioritizing project claims as opposed to individual downsides. The third question asks students to rank the key factors considered in the model in order of

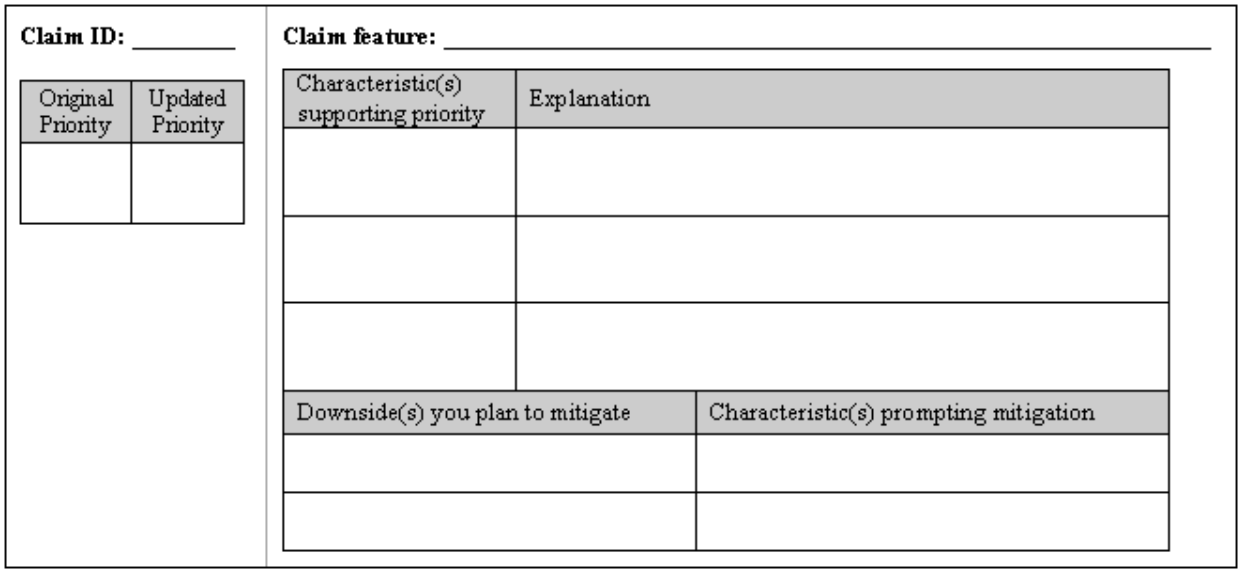

Figure 5.6: Chart summarizing risk-related characteristics of a project claim

importance for estimating risk, and the fourth question probes teams' plans to coordinate their redesign efforts by assigning mitigation tasks to individual team members.

Following the redesign phase of the project, the teams were asked to report on the specific downsides they mitigated as well as provide rationale for why they chose not to mitigate some high-priority downsides. For each claim in their project set in which one or more downsides were mitigated, the teams were asked to complete the chart depicted in Figure 5.7.

The detailed specifications for the two risk-related assignments given to these student teams are presented in Appendix B and Appendix C. The results of the evaluation are presented in the next section.

Claim ID: _____ **Original Priority:** _____

Claim feature: _____

Downside(s) mitigated	Claim mitigating this downside	
	ID	**Feature**

How does mitigation of this downside(s) improve your interface?

Figure 5.7: Chart used by design teams to articulate particular downsides mitigated during redesign

5.4 Results

Twenty-one design teams participated in the evaluation; however, several groups were removed from the evaluation for various reasons. Out of thirteen original teams in one section of the course, one team failed to turn in their assignment, one group did not list original priorities for their project claims, one team updated multiple claims to the same priority, and two teams failed to update the priority of any of the claims in their project sets. Since all other teams updated the priorities of multiple claims, it was assumed that these two teams did not take the assignment seriously. Out of nine original teams in the second section of the course, one team provided original and updated priorities greater than the number of claims in their project set, and one team did not use the provided characteristics in support of their updated priorities.

The results acquired from these seven teams were removed from the evaluation, which left the results of eight teams using the stakeholder concern model and the results of seven teams using the comprehensive risk model to be analyzed. The analysis of data collected in response to the six research questions outlined above is presented in the following sections.

5.4.1 Benefit of Prioritizing Claims versus Downsides

The frit research question involves the benefit of prioritizing project claims as opposed to individual downsides. Although individual project downsides represent actual design risks, the risk models presented here prioritize project claims, as discussed in Chapter 3. Following the prioritization of their project claims, teams were asked about the perceived benefit of prioritizing claims in addition to the potential benefit of prioritizing individual downsides. In each case, the teams were presented with a five-point Likert scale of possible responses.

Figure 5.8 depicts the distribution of responses to the benefit of prioritizing claims. Out of the 15 project teams surveyed, 11 teams felt that claim prioritization was somewhat to extremely beneficial. These responses were distributed evenly between teams in each section of the course.

Several teams mentioned that claim prioritization improved the process of redesign and helped to identify critical issues that should be addressed, or at least given consideration, within the context of the project. Several teams stated that prioritization helped them to organize and plan the steps of their redesign and remain focused on the most important design issues without being slowed down by less-critical downsides. One team also mentioned that it is important for designers to set small goals and that claim prioritization aided designers in choosing the best subset of those goals that would ultimately allow them to reach their overall design goals.

However, several other teams stated that the process of prioritization, be it claims or individual downsides, was somewhat tedious. One team mentioned that downside mitigation would be most beneficial if it was an optional feature within LINK-UP. This team wanted the ability to select a subset of their project downsides to prioritize. Although this option might be beneficial to experienced designers, or to novice designers working on small-scale projects, key goals of integrating a risk model within HCI and LINK-UP is to automate the process whenever possible and to draw attention to critical issues that might otherwise have gone unnoticed.

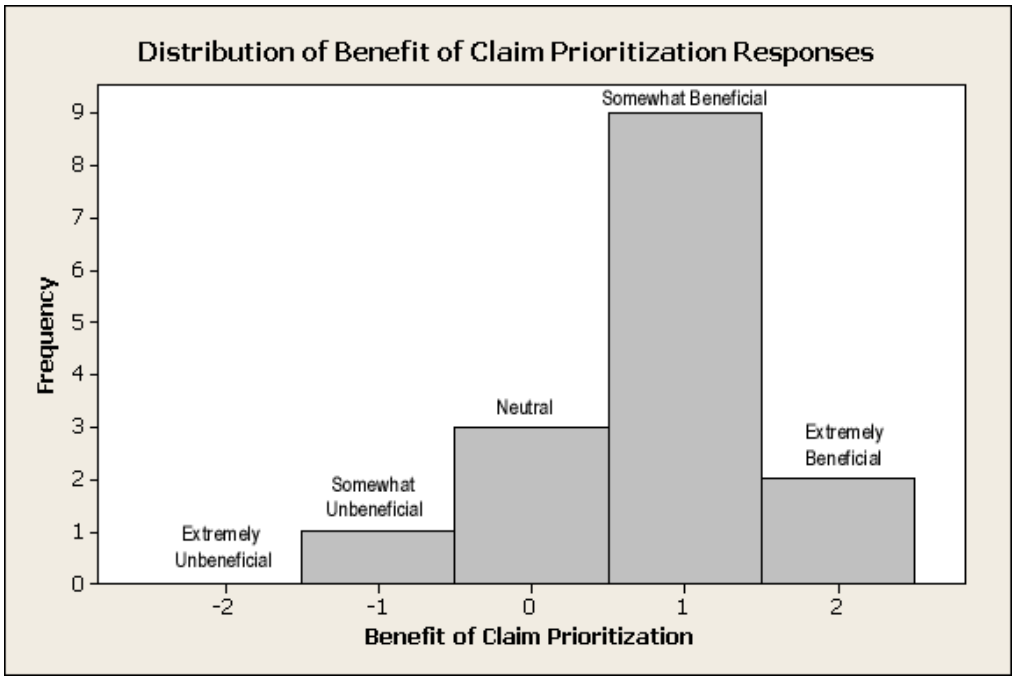

Figure 5.8: The distribution of design team responses to the benefit of claim prioritization

Project teams that strongly disagreed with the results of their prioritization also felt that, although the process was useful, the accuracy of the model was an issue to be addressed. Time spent reprioritizing their list of project claims would have been better spent moving forward with the redesign of their system.

Figure 5.9 depicts the distribution of responses to the benefits of prioritizing individual downsides. In this case, teams were far less certain about the benefit of prioritization. Seven out of 15 teams felt that downside prioritization would be somewhat to extremely beneficial. Five teams were neutral to the idea of downside prioritization, one team felt that it would be somewhat unbeneficial, and one team felt that it would be extremely unbeneficial.

Several teams felt that downside prioritization would provide a clearer picture of the problems associated with a system, allowing designers to view the system as a whole and address issues more efficiently. However, other teams recognized that downside prioritization was done as a part of claim prioritization, and therefore, felt that individual downside prioritization would be overkill. Given the number of possible downsides in a project, the process would become even more tedious.

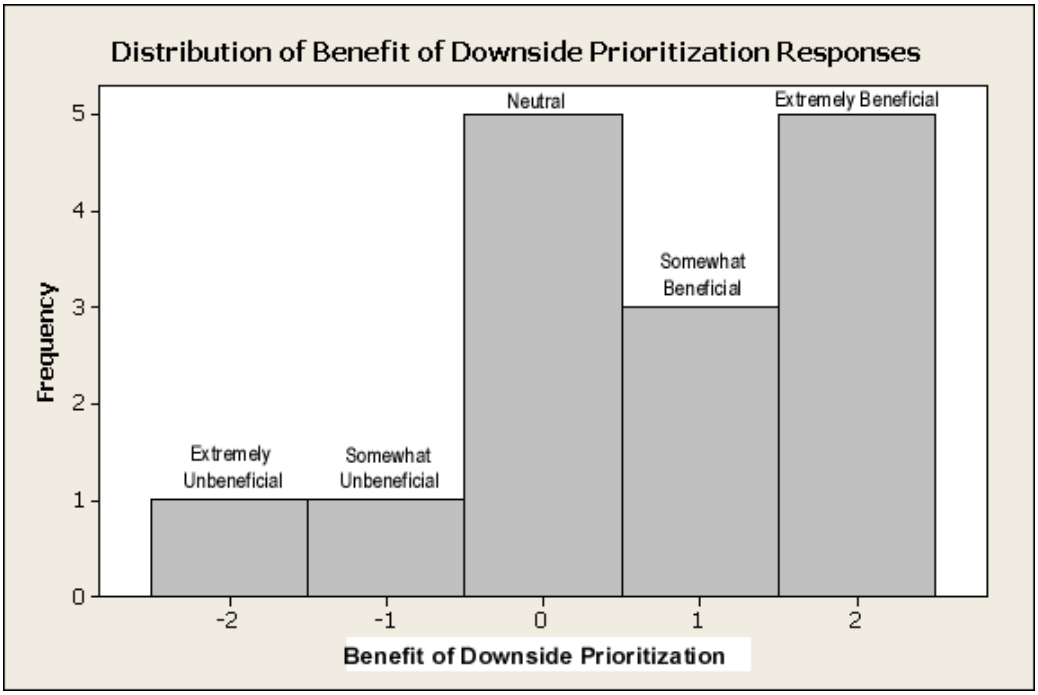

Figure 5.9: The distribution of design team responses to the benefit of downside prioritization

These results, acquired from undergraduate, novice designers, might reflect student's desire to complete the simplest procedure possible, and since claim prioritization seems less tedious to them than individual downside prioritization, they see greater benefit in the former. Additionally, these results might suggest different preferences in the type of prioritization based on the designer in question. Perhaps novice designers prefer claim prioritization because it aids in organizing design knowledge and maintaining design context. In contrast, more experienced designers might prefer downside prioritization, which provides a more streamlined method determining which design risks to address. Additional user studies, involving more experienced, professional designers, are needed to confirm or deny these speculations.

5.4.2 Contribution of Risk Management to System Redesign

The second research question asks how well the two risk models contribute to the redesign phase of a project. After assessing their prioritized claim lists, the project teams chose several downsides that they planned to mitigate during redesign. Following the redesign phase, teams reported on the specific downsides that they actually did mitigate during a brief, one-week redesign phase. The mean original priority of the claims in which these mitigated

downsides appeared (i.e., the claim priority determined by each risk model before teams reprioritized the lists) was calculated and compared to the grade each team received for the redesign phase of the project.

The phase grades consider only the quality of the system redesign and did not take into account the degree to which each team followed instructions and completed the project as assigned. Grading criteria included the quality of the teams' explanations for how each downside they chose to mitigate improved their interface, the quality of their justification for why certain downsides were not mitigated, and the degree to which the project grader felt the overall design improved as a result of the changes made. In this case, grades for the redesign phase of the project represent the degree to which each project design improved as a result of mitigating a subset of downsides.

Simple regression analysis was performed to determine the correlation between original risk priority of mitigated downsides and resulting project grades, where mean priority of mitigated downsides is the predictor variable and phase grade is the response variable. The null hypothesis is that teams that choose to mitigate higher priority downsides will earn higher grades for their redesign. To rule out the possibility of differences in grades between the two sections of the course based on extraneous factors, grades from two previous project phases were compared between course sections. Separate two-group t-tests for each of the two project phases yielded no significant difference between the mean grades for each course section ($p = 0.561$ and $p = 0.747$, respectively).

The results from two of the 15 teams were considered outliers (one with a significantly lower phase grade and one with a significantly lower mean priority than the remaining teams) and removed from the analysis. Although the regression was a poor fit ($R^2_{adj} = 32.6\%$), the overall relationship between variables was significant ($F_{1, 10} = 6.31$, $p = 0.031$). A significant P-value, in this case, shows that the model as a whole does aid in understanding the relationship between original priority of mitigated downsides and quality of the resulting design product. Priority of mitigated downsides is a significant predictor of phase grades ($t_{10} = 2.51$, $p = 0.05$). For the given single-factor regression model, this t-value serves to reinforce the significance of the overall model. Although the model does not account for all of the variance, it does show that teams that mitigate downsides from high-risk claims achieve a somewhat higher quality interface.

In addition to empirical data collected during the evaluation, teams were also asked a follow-up question to determine the degree to which they followed the priority list during the redesign phase of the project and the extent to which they felt the priority list was a useful guide to the redesign of their interface. In each case, the teams were presented with a five-point Likert scale of possible responses.

First, teams were asked about the degree to which they followed the prioritized claim list during the redesign of their interface. Possible responses include:

1. We did not consider claim priority at all when determining which downsides to mitigate
2. We considered claim priority during redesign, but the ultimate decision to mitigate a downside was made regardless of claim priority
3. We referred to the priority list throughout redesign and mitigated downsides from high priority claims whenever possible
4. We followed the list closely and refused to mitigate downsides from claims that were not in the top one-third of the priority list

Figure 5.10 depicts the responses of 15 project teams. The results show that all teams at least considered the priority list as they redesigned their interface. Most teams (10 out of 15) referred to the priority list frequently and attempted to mitigate downsides from high priority claims whenever possible.

Figure 5.11 depicts the same 15 teams' responses to the degree to which the priority list was useful in guiding the redesign phase of the project. Possible responses range from 'of no use' to 'extremely useful.' These results show that the majority of teams (10 out of 15) felt that the priority list was somewhat useful in guiding project redesign. One team felt that the priority list was extremely useful, while four other teams were neutral to the benefit of the list. No teams felt that the priority list was useless. Several teams commented on the guidance of the priority list in terms of organization and focus on key design issues. Many teams attempted to address the high priority

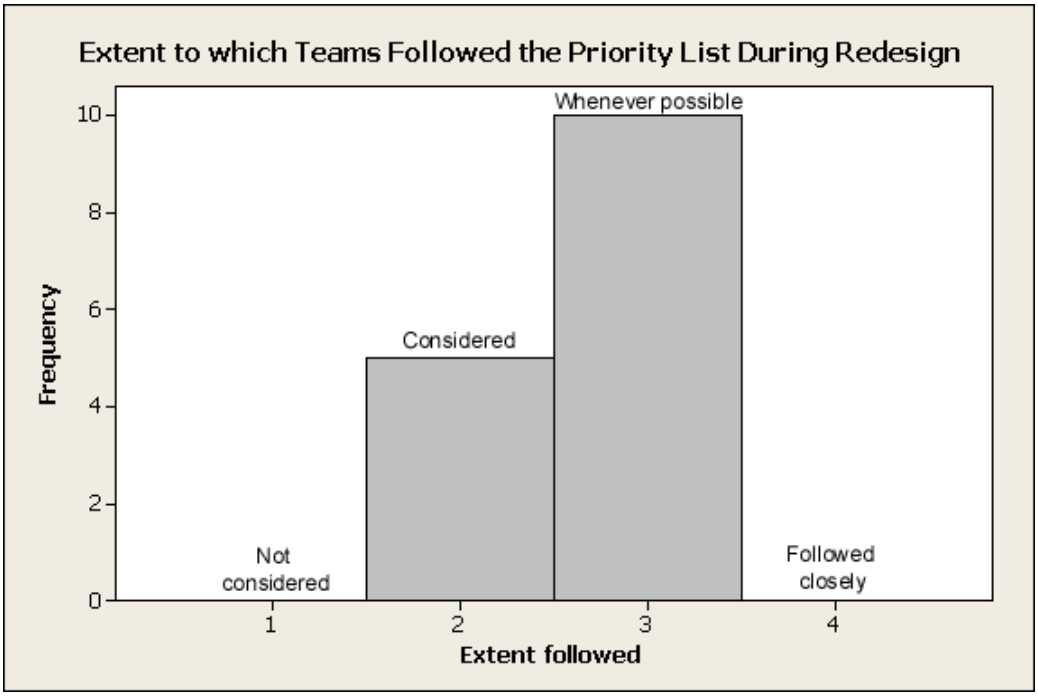

Figure 5.10: Extent to which priority lists were followed during redesign

downsides in their design; however, some mentioned problems such as not having enough time to mitigate some of the critical but more complex downsides. As a result, they were forced to ignore these design issues. However, most teams stated that the priority list provided useful guidance during the redesign process even if they did not follow the priority order exactly.

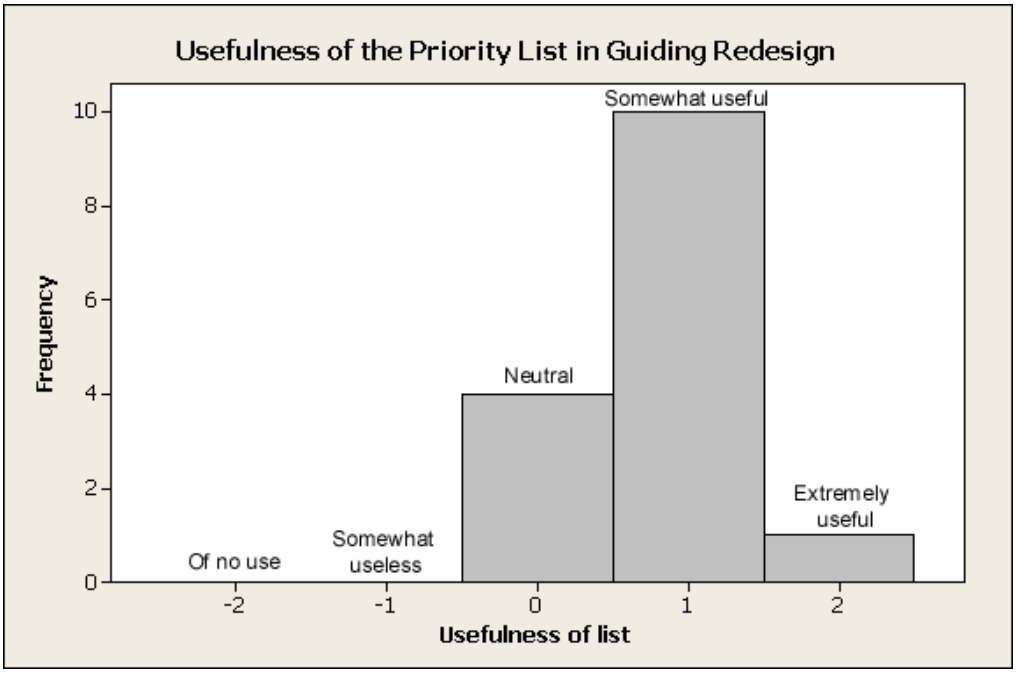

Figure 5.11: Usefulness of claim priority list during redesign

5.4.3 Contribution of SBD to Risk Management

The third research question seeks to determine the benefit of SBD to risk management in terms of facilitating communication about design issues. Following the initial design phase of the project, the design teams acted on behalf of project stakeholders to assign concern values to each of their project downsides. Following this activity, they were asked about how the use of scenarios and claims affected the ease with which their team was able to discuss the design of their system. The teams were presented with a five-point Likert scale of possible responses ranging from 'use of scenarios and claims made stakeholder discussions significantly easier' to 'use of scenarios and claims made stakeholder discussions significantly harder.' All 21 teams completed this initial assignment correctly; thus, all of their responses to this follow-up question are included here. The results are summarized in Figure 5.12.

Teams were also asked to provide a brief explanation for their response. In particular, they were asked to discuss what characteristics of scenarios and claims contributed to or detracted from their discussions as project stakeholders. More than 50 percent of the teams surveyed (14/21) stated that use of scenarios and claims made design discussions somewhat to significantly easier. Many of these teams felt that scenarios and claims drew out design details that they might not have otherwise considered. Scenarios and claims helped several teams to narrow the focus of their design and identify new perspectives of their system. They allow teams to maintain an overall view of the design project while focusing on one aspect of the design at a time.

Scenarios provide the background of a real-world situation in which the system might be used, while claims draw attention to potential problems that need to be addressed. Additionally, upside and downside rationale aided several teams in discussing the relevance and worth of each individual claim. By condensing rationale within the

structure of a claim, teams do not waste time searching for rationale to support their claims. This time can be better spent discussing the claim itself in relation to the design as a whole.

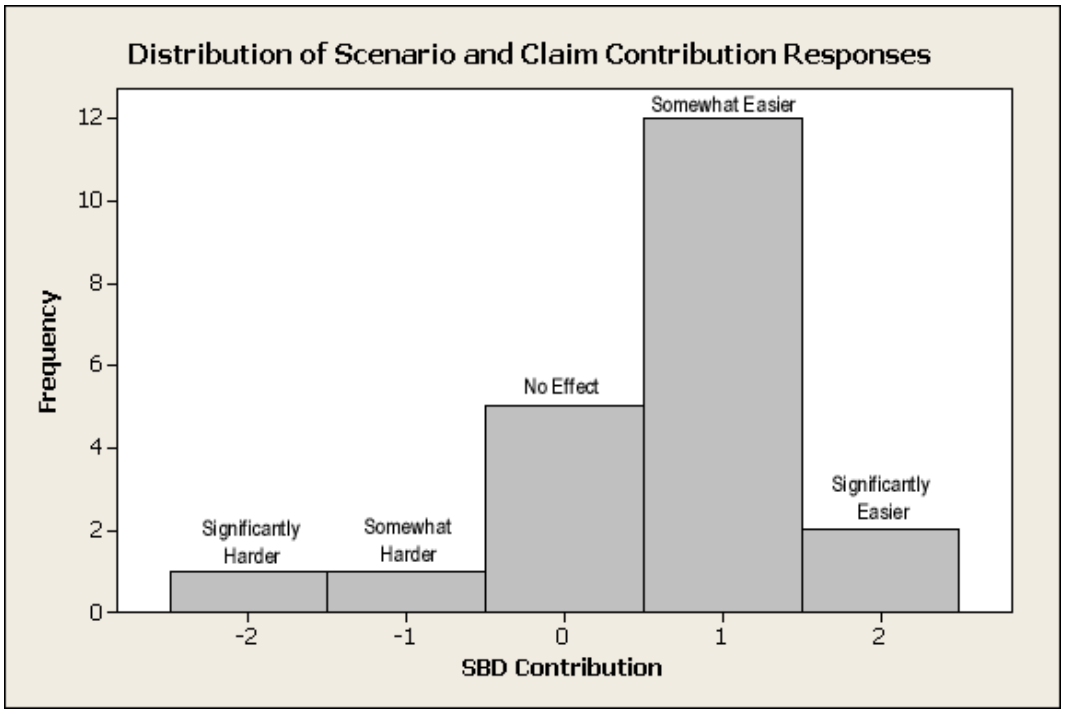

Figure 5.12: Distribution of scenario and claim contribution responses

However, not all teams agreed that scenarios and claims were beneficial to stakeholder discussions. As novice designers, some teams had trouble using the terminology associated with scenario-based design and felt that they had not been using the process long enough to think clearly in terms of scenarios and claims. One team also commented that, since they, as the system designers, were already familiar with their design, scenarios and claims were not as useful in their discussions as they might have been if external project stakeholders were also involved.

Several teams stated that scenarios and claim had no effect on their group discussions but then mentioned that claims and scenarios have both good and bad qualities. For example, some teams felt that, although claims and scenarios improved communication among project stakeholders and aiding in the organization of design knowledge, the process of creating a claims analysis record was somewhat tedious and sometimes distracted them from their overall design goals. One team stated that claims and scenarios had no effect on their design process because the design of their small-scale system was fairly straightforward. Some groups also mentioned that, while scenarios were helpful in guiding discussion about a design, claims were overly complicated when applied to the small-scale projects on which they were working.

5.4.4 Facilitation of Task Allocation

The fourth research question asks whether use of the risk model aids designers in identifying and allocating key project tasks. Each downside chosen for mitigation during the redesign phase of a project represents a project task that could be assigned to one or more team members. The student design teams participating in the evaluation were asked if they planned to approach the redesign phase of their project by assigning responsibility for individual downside mitigation to specific team members. Possible responses were 'yes' or 'no,' and the teams were then asked to explain their response. Only two out of the 15 teams responded that they were planning to allocation individual mitigation tasks to individual team members. The other 13 teams preferred to work together as a group.

The two teams that did choose to assign individual tasks felt that it would improve efficiency and time management, given the short time frame in which they had to complete their redesign. One team also mentioned that working individually and then presenting work to the group would allow for counter checks of each team member's work, as opposed to the entire team working together toward the same goals.

The teams that chose not to assign individual mitigation tasks cited several reasons for their decision. A number of teams discussed the relationships between claims and downsides that might negatively impact the overall system design if mitigations were done separately. By working together as a group, the designers can ensure that each mitigation has only a positive effect on the design as a whole. Teams also mentioned that working together as a single group facilitated creativity and discussion of design ideas. As a result, everyone on the team had a greater chance of being heard, and conflicts could be resolved sooner and with greater efficiency. Additionally, working together helped some teams to answer questions about the assignment itself and to ensure that everyone on the team was working toward the same goal.

Several teams also mentioned that the small size of their project teams did not inhibit them from working efficiently as a single group. However, they did indicate that individual task allocation would be of benefit for a larger project team. Given a more complex design project and a larger and/or more distributed project team, allocation of tasks based on individual downside mitigation, or on subsets of related downsides, might be more beneficial. However, additional evaluation is needed to determine this effect on larger teams.

5.4.5 Extent of Designer Agreement with Risk Prioritization

The fifth research question seeks to determine the extent to which designers agree with the prioritization of project claims based on each of the two risk models used. An accurate risk model must push the most critical design risks to the top of the list, and this question serves to test the accuracy of the models presented here.

This question also seeks to determine the degree to which designers trust each of the two models. The concept of a simpler, one-factor risk model, such as the stakeholder concern model, is appealing in many ways because the use of a single factor allows the user to determine more easily how claim priorities are determined. As a result, the model makes more sense to the user and, presumably, will gain more of the user's trust. In contrast, a model that considers multiple factors, such as the comprehensive risk model, is more difficult for the user to decipher and, thus, might earn less of the user's trust.

After teams examined the original priority list of their project claims produced from LINK-UP, they were asked to reprioritize the claims as they saw fit. Although they were told that they could choose to increase, decrease, or maintain the priority of each claim, most teams chose to update the priorities of the majority of their claims and maintain the priorities of only a few of the claims in their list. These heavy reprioritizations might reflect the students' desire to complete the assignment sufficiently as well as their genuine disagreement with the prioritization results.

Designer agreement was measured according to four separate factors: the percentage of claims that were assigned new priorities per team, the mean change in priority per team, the maximum change in claim priority per team, and the number of claims per team that remained in the top 5 of the priority list after the list was updated. In each case, the null hypothesis is that there is no difference between the means for each model group. An analysis of each of these factors is presented below, and the results are summarized in Figure 5.13.

The percentages of claims that were assigned new priorities per team were calculated by dividing the number of claims with an updated priority by the total number of claims in each project set. The percentages are listed in Table 5.8-A. Although the overall mean percentage within teams using the stakeholder concern model is somewhat lower than the mean percentage within teams using the comprehensive risk model, a statistical comparison of the means yielded no significant difference between the two groups ($t_6 = 1.30$, $p = 0.240$).

The mean change in claim priority per team was calculated by summing the change in priority for each individual claim and dividing by the total number of claims in the project set. The mean for each team is presented in Table 5.8-B. In this case, the means for the two model groups are extremely close, and no statistical difference was detected ($t_{12} = 0.08$, $p = 0.940$).

The maximum change in claim priority for each team is presented in Table 5.8-C. In this case, the mean maximum difference in priority within teams using the stakeholder model is somewhat higher than the mean maximum difference within teams using the comprehensive model. This result shows that, although a higher percentage of claim priorities were changed by users of the comprehensive model, the size of the change was often smaller. However, the difference between the two means was not significant ($t_{11} = 0.95$, $p = 0.363$).

The number of claims remaining in the top 5 of the priority list was measured by comparing the original and updated priority lists for each team and counting the number of claims that appeared in the top 5 for both lists. The results are presented in Table 5.8-D. Although the mean within teams using the stakeholder concern model is slightly higher than the mean within teams using the comprehensive risk model, the difference between the two means was not significant ($t_{10} = 0.60$, $p = 0.562$).

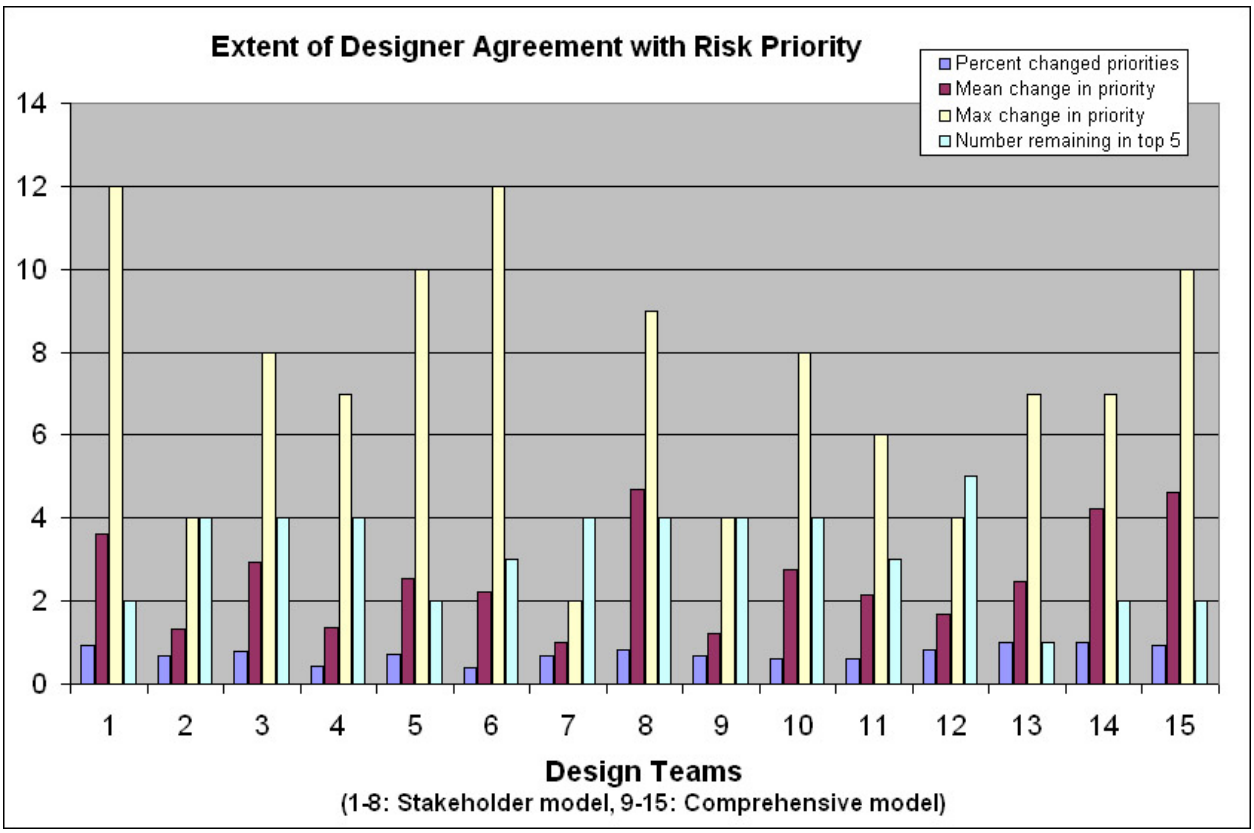

Figure 5.13: Summary of extent of designer agreement measures

Based on these results, the null hypothesis cannot be rejected. In terms of accuracy, this result shows no significant difference between the levels of performance of each model. However, these results do suggest that users trust both risk models. Although the stakeholder concern model might be more appealing to users in terms of simplicity and ease of understanding, these results show that users are willing to trust the results of a more comprehensive model, which, with future improvements, might also provide a more accurate estimation of risk.

Percentage of Changed Claim Priorities per Project Team	
Stakeholder Model	**Comprehensive Model**
0.93	0.67
0.67	0.62
0.80	0.62
0.44	0.83
0.73	1.00
0.41	1.00
0.67	0.92
0.82	
Mean: 0.68	**Mean:** 0.81
Std Dev: 0.181	**Std Dev:** 0.172
P-value = 0.240	

A. Percentage of changed priorities

Mean Change in Claim Priority per Project Team	
Stakeholder Model	**Comprehensive Model**
3.600	1.200
1.333	2.769
2.933	2.154
1.375	1.667
2.533	2.483
2.235	4.222
1.000	4.615
4.706	
Mean: 2.464	**Mean:** 2.520
Std Dev: 1.265	**Std Dev:** 1.268
P-value = 0.940	

B. Mean change in priority

Maximum Change in Claim Priority per Project Team	
Stakeholder Model	**Comprehensive Model**
12	4
4	8
8	6
7	4
10	7
12	7
2	10
9	
Mean: 8.000	**Mean:** 6.571
Std Dev: 3.590	**Std Dev:** 2.149
P-value = 0.363	

C. Maximum change in priority

Number of Claims Remaining in Top 5 after Update	
Stakeholder Model	**Comprehensive Model**
2	4
4	4
4	3
4	5
2	1
3	2
4	2
4	
Mean: 3.375	**Mean:** 3.000
Std Dev: 0.916	**Std Dev:** 1.414
P-value = 0.562	

D. Number remaining in top 5

Table 5.8: Measurements of extent of designer agreement with prioritization results

5.4.6 Critical Factors in Design Risk Estimation

The final research question seeks to determine what factors are most important in an estimation of design risk. The five key factors considered in the comprehensive risk model are stakeholder concern, results of internal evaluation, quality and relevance of external rationale, overall claim quality, and difference in claim and system IRC values. After completing the claim prioritization aspect of the assignment, the student design teams were asked to rank these five factors in order of importance for estimating design risks. They were also encouraged to add and rank additional factors that they felt were important. The results of this ranking exercise are presented in Table 5.9.

Team Rankings of Risk Factor Importance					
	Claim Quality	IRC Difference	Stakeholder Concern	External Rationale	Internal Evaluation
Stake Mean	3.750	3.625	1.125	3.625	2.875
Comp Mean	3.000	4.286	1.143	3.714	2.857
Overall Mean	3.400	3.933	1.133	3.667	2.867
Overall Std Dev	1.056	1.033	0.352	1.397	1.060
Num 1st Rank	0	0	13	1	1
Num 2nd Rank	3	2	2	3	5
Num 3rd Rank	6	2	0	2	5
Num 4th Rank	3	6	0	3	3
Num 5th Rank	3	5	0	6	1

Table 5.9: Mean rankings for each of the five key risk factors considered in the comprehensive risk model

Stakeholder concern is obviously considered the most critical risk factor, with 13 out of 15 teams ranking it as the most important factor and the remaining two teams ranking it as the second most important factor. Since the student design teams themselves were responsible for assigning stakeholder concern ratings to the claims and downsides in their project set, these results might reflect, to some degree, the amount of effort that students expended in completing this task. However, the student design teams were also involved in determining a number of other risk factors, including internal evaluation results and the factors contributing to claim quality; thus, the importance of stakeholder concern ratings should not necessarily be biased by degree of designer input. Additionally, the results of the OE case study confirm the importance of these user-influenced factors.

The remaining four risk factors are not ranked as consistently by the 15 design teams. Based on overall mean, internal evaluation is ranked as the second most important factor, followed by claim quality, external rationale, and IRC difference. However, based on the spread of ranking throughout the 15 teams, these four factors are given varying degrees of importance by various groups.

5.5 Summary of Results

The purpose of this evaluation was to answer six key research questions, as discussed in Section 5.1. First, teams were asked to comment on the benefit of prioritizing claims as opposed to individual downside risks. The majority of teams stated that prioritization of claims was somewhat beneficial. Teams were less sure of the benefit of prioritizing downsides. A number of teams took a neutral standpoint, and two teams felt that downside prioritization would not be beneficial. However, approximately half of the teams surveyed felt that downside prioritization would be somewhat to extremely beneficial. Terrell also speculated that she, as a designer, would prefer downside prioritization, provided that the downside was explicitly linked to its associated claim. Additional evaluation is needed to compare the actual benefits of prioritizing claims versus individual downsides.

To determine the contribution of a risk model to the redesign process, teams were asked to choose a subset of project downsides to mitigate during the redesign phase of their semester project. The mean priority of claims containing downsides actually mitigated during redesign was compared to teams' resulting phase grades. Although

the regression was a poor fit, the overall regression model is significant, showing that teams that mitigated higher priority downsides did earn somewhat higher grades for their redesign.

When asked how the use of scenarios and claims contributed to design discussions among project stakeholders, the majority of design teams felt that SBD made those discussions somewhat easier. Nevertheless, several teams commented on the complexity and tediousness of the process for novice designers, as described in Section 5.4.4.

Teams were also asked whether they planned to allocate individual downside mitigation tasks to individual team members. Only two teams replied that they did intend to allocate mitigation tasks, while the remaining teams stated that they preferred to work together as a single group. Several teams mentioned that the small size of their project group influenced their decision and that a larger project team would benefit more from task allocation. Additional studies of larger project groups are needed to determine the benefit of mitigation task allocation.

To determine the extent to which designers agree with and trust the results of both the stakeholder concern model and the comprehensive risk model, student design teams were asked to reprioritize the list of their project design claims, resulting from one or the other of those models, as they saw fit. Results yielded no significant difference between the mean percentage of changed priorities, the mean change in priority, the mean maximum change in priority, or the mean number of claims remaining in the top five of the list after reprioritization. The lack of significance might be due, in part, to the small sample size of the two groups. However, retention of the null hypothesis does suggest that designers trust both the comprehensive model and the simplified stakeholder concern model. Although the student design teams participating in evaluation judged the accuracy of both models similarly, Terrell, the lead designer on the OE project, felt strongly that the priority list resulting from the comprehensive risk model more accurately estimated design risk. Although the OE results consider only one system, Terrell was given a much more detailed explanation of the purpose and composition of the design risk model and spent more time formulating her responses.

Finally, to determine what factors are most important to consider in estimating design risk, the teams were asked to rank the top five components considered in the comprehensive model in terms of importance for estimating risk. Stakeholder concern was selected almost unanimously as the most important risk factor. Although the remaining four factors were ranked differently by various design teams, based on the mean rank for each factor, results of internal evaluations was ranked as the second most critical risk factor, followed by claim quality, external rationale, and IRC difference. Feedback from Terrell supported these results. She also ranked internal evaluation results and stakeholder concern as the top two factors, stating that user-driven data is far more useful than designer-driven data in terms of estimating design risk. A summary of evaluation results is presented in Table 5.105.10.

Claim prioritization is beneficial to the redesign process	✓
Claim prioritization is more beneficial than downside prioritization	?
Scenarios and claims support design discussions and facilitate reuse of risk-related knowledge	✓
Risk model facilitates task allocation for small-scale teams	X
Risk model facilitates task allocation for large-scale teams	?
Designers trust results of both risk models	✓
Prioritization results of the comprehensive model are more accurate then results of the stakeholder concern model	?
User-influenced risk factors are more critical than designer-influenced factors	✓
Designer-influenced factors are still important for an accurate assessment of design risk	?

Table 5.10. Summary of the results of the risk model evaluation

The results of this evaluation show that risk management is a beneficial concept within the HCI domain. Nearly all teams participating in the study acknowledged that use of the risk model helped them to identify and focus on the most critical design risks and organize their plans for redesign more efficiently. The initial version of the comprehensive model performed comparably with the stakeholder concern model and was accepted equally by the designers. Improvements to the comprehensive model, based on feedback from designers in terms of which risk factors are most important, could lead to a more accurate model that is trusted by designers and useful to design teams.

Chapter 6 : Conclusions

As software-intensive projects continue to grow in size and complexity, a successful design process becomes more critical. An increase in the size and distribution of project teams also increases the need for successful project management paradigms. Since most management tasks add significant overhead to a project, they are often overlooked and, as a result, can lead to project failure. To reduce overhead, management tasks must be integrated directly into the design process.

All projects have risks, and lack of emphasis on risk management is a key downfall of successful project management. The concept of risk coincides nicely with the idea of a claim, and the integration of risk management within the Scenario-Based Design process serves not only to improve management techniques in design projects, but also to further the science of design. To evolve into a true engineering discipline, HCI must gain respect through the development of a more scientific approach to interface design. This scientific approach involves understanding and articulating the steps needed to achieve quality design, accounting for the goals of system stakeholders, and automating error-prone human activities whenever possible [41]. The systematic process of identifying, assessing, prioritizing, and controlling project-related risk contributes to this quest for a true design science. Use of a risk model adds structure to the redesign process, guiding designers through multiple iterations when the traditional SBD process falls short. The risk model considers stakeholder goals through the incorporation of stakeholder concern ratings and the results of design evaluations. Implementation of the risk model within LINK-UP simplifies the process of inputting risk-related knowledge and automates risk calculations, reducing the probability of human error. The integration of a risk model within the SBD process also adds an element of project management to the science of design, helping collaborative design teams to remain aware and to work together more effectively.

This thesis begins to address improvements to both the science of design and the management of collaborative projects through the following contributions:

- A review of existing collaborative tools, resulting in a set of guidelines for project management tool support in collaborative environments
- An adaptation of existing techniques for management of process-related risks into a new domain to address management of product-related risks in HCI design
- An initial, proof of concept, comprehensive risk model integrated within an existing design process, which allows designers to manage design risk with minimal project overhead
- Implementation of the risk model within the LINK-UP design environment, which simplifies the process of estimating risk by automating risk calculations based on information being used in other aspects of the design
- Results of an initial user evaluation, which provide insight into the benefits and shortcomings of managing claims as design risks

The next section discusses several key lessons learned through the development and evaluation of risk management within HCI design. Section 6.2 then outlines critical directions for future work.

6.1 Lessons Learned

Results of the development and evaluation of this work show that risk management is a useful concept within the context of HCI design. Nearly all teams participating the in the evaluation, regardless of the model used in assessing their design risks, acknowledged the benefit of risk prioritization toward the redesign of their interface. However, this initial evaluation did not definitively determine the benefit of prioritizing claims as opposed to individual downsides. A large percentage of project teams, as well as Terrell, the lead designer on the OE project, reacted more naturally to the notion of prioritizing individual downside risks. Nevertheless, all of these teams, as well as Terrell, acknowledged the need to maintain a strong link between an individual downside and the context of its associated claim.

Additionally, the majority of teams acknowledged the benefit of scenarios and claims in facilitating design discussions among designers and other project stakeholders. The use of natural language scenarios to present usage situations, as well as the use of claims to extract and summarize key design features and their associated tradeoffs, helps stakeholders to discuss the design of the system and, consequently, to understand and assess possible risks. By involving multiple project stakeholders in the risk management process, risk assessment incorporates different perspectives, eliminating bias and resulting in a more diverse review and a more accurate risk prioritization.

The risk model also supports knowledge reuse through the use of claims. Whenever possible, risk-related values are stored within the structure of a claim at a high level, allowing this information to be input into a knowledge repository and reused by multiple designers working on various projects. The claim quality rating, IRC difference factor, and numerous factors related to each source of external rationale supporting the claim can be input into LINK-UP by the claim author and then archived with the claim for reuse. As a particular claim in reused in multiple project settings, the claim quality rating with be updated to reflect the success or failure of that claim. In this way, the claim quality rating encourages the reuse of quality knowledge. The degree to which a claim has been reused is one factor considered in calculating claim quality, causing reuse to operate in a cyclic fashion. High quality claims are more likely to be reused, and successful reuse is then rewarded with a higher claim quality rating.

Reluctance of project teams to allocate individual tasks based on mitigation of project downsides was a surprising, yet, in hindsight, logical result. The majority of teams participating in the evaluation chose to work as a single unit throughout the redesign phase of their project. However, teams cited group size and project simplicity as key factors in their decision. Teams of three to four students designing a small-scale notification interface had little need to divide their efforts. Furthermore, these students relied heavily on teamwork to understand the design concepts presented in class, to understand the assignments they were given, and to coordinate their knowledge toward a common goal. Several teams explicitly stated that larger groups of designers working on more complex projects would be more likely to benefit from individual task allocation. Thus, the initial hypothesis, namely that use of the risk model would facilitate task allocation, was neither verified, nor rejected, through this initial evaluation of

the risk model. Additional studies with larger project teams are needed to determine the benefit of task allocation based on mitigation of downside risks.

The evaluating designers' rankings of key risk factors agreed with initial predictions. Specifically, stakeholder concern ratings and internal evaluation results were considered the most important considerations in an accurate estimation of risk exposure. Future versions of the risk model should focus on these key factors; however the remaining factors should not be ignored. Additional research is needed to determine exactly how these other factors, such as claim quality and difference in IRC ratings, actually contribute to the accuracy of risk assessments. For example, this initial risk model assumes that a claim with an IRC rating closer to the design model IRC rating is more likely to contribute to the design goals of the system. However, the exact relationship between claim and design model IRC ratings is unknown. A more precise understanding of this relationship could lead to a more accurate representation of this factor in future versions of the risk model.

In some cases, the knowledge needed to calculate risk exposure values for each claim in a project set is somewhat cumbersome to collect. Although each piece of knowledge relates to the design phase in which it is collected, the need to specifically assign numerous values to each individual downside of each project claim can cause designers to lose focus on key design tasks. As the size of the project and the associated scenario and claim sets continues to increase, scalability of the risk model suffers. Future versions of the model must attempt to collect risk-related knowledge in a way that contributes more thoroughly to the design task at hand and also helps the designer to understand that contribution. As a result, risk management will emerge more naturally from the design process, contributing to design improvements as opposed to distracting designers from their overall goals.

Although the evaluation results do not show more accurate predictive power using the comprehensive model as opposed to the stakeholder concern model, they do suggest that designers trust both the calculation process for a complex, multi-factor model as well as the calculations for a simpler, single factor model. Since designers trust the comprehensive risk model, with increased accuracy, the comprehensive model has the potential to provide a more beneficial assessment of design risk than a simpler, more static model. Furthermore, Terrell stated that the results of the comprehensive model were already more accurate than those of the stakeholder concern model. Although these results reflect the opinion of a single designer within the context of a single system design, they do represent the opinion of a more experienced designer who took her time in examining and comparing the results of both risk models. In contrast, the undergraduate HCI students who participated in the user evaluation were novice designers who potentially rushed through the assignment without a clear vision of when and how the risk model should be used. Additionally, these student evaluators had access to the results of only one model. Instead of comparing the results of both models, they simply reprioritized their project claims as the assignment stated. More in-depth evaluations, with larger sample sizes of both novice and experienced designers, are needed to fully assess the accuracy of the risk models.

The integration of risk management within HCI design currently benefits the design of notification systems. However, to truly contribute to the advancement of HCI and the science of design, the risk model must be expanded for use in other design domains. Provided that design processes in other domains adopt the use of claims, the general structure of the risk model and the majority of its risk factors can be reused. For example, the definitions of claim

quality, stakeholder concern, and results of internal evaluations can be transferred directly from one domain to another. Only critical parameters and rationale relevance factors will need to be redefined for each new domain. Based on existing research on the use of critical parameters [38], the definition of these parameters for other design domains will significantly benefit the design process in addition to facilitating use of this risk model.

The initial comprehensive risk model draws not only upon existing risk management practices in other domains, but also upon the need for a science of design. The overall goal of the model is to inject risk management into the design process without adding significant overhead to the project. The lessons learned through the development and evaluation of this initial risk-driven, claims-centric management model can be summarized as follows:

- **Risk management benefits HCI design by focusing attention on critical design issues**
  Within the context of scenario-based notification systems design, prioritization of project claims according to degree of risk exposure aids designers in identifying and focusing on critical design issues. The comprehensive risk model considers knowledge gained through previous design endeavors, goals of the project stakeholders, and the existence of problems as supported by the results of internal design evaluations. Additionally, the majority of risk factors considered in the comprehensive model can be reused across domains, facilitating the transfer of risk management into other areas of design.
- **Scenario-Based Design supports management of design risk by structuring risk-related knowledge for discussion and reuse**
  The key stages in the SBD process support risk model integration, improving design and management processes with minimal added overhead. The structure of scenarios and claims, which are the key components of SBD, support design discussions among project stakeholders and facilitate reuse of risk-related design knowledge.
- **A comprehensive model is advantageous to design risk management because consideration of all available information results in a more thorough risk assessment**
  Although a single-factor risk model is easier for the designer to understand, evaluation results show that designers also trust the results of more complex risk calculations. Use of all available information allows risk to be estimated much earlier in the design process and results in a more comprehensive risk assessment. A comprehensive model has the potential, with improved predictive power, to provide a more accurate and beneficial assessment of design risk than a simpler, more static model.

Based on these lessons learned, the next section elaborates on several key directions for future work.

6.2 Future Work

While the work contained within this thesis provides a strong beginning to the integration of risk management within HCI design, it provides only an initial version of a risk-driven, claims-centric management model. The comprehensive risk model presented in Chapter 4 was sufficient for use in accomplishing the overall goals of this thesis –to establish the benefits of integrating risk management within an HCI design process and to determine the

key factors to consider in estimating design risk. However, future work is needed to reassess and improve the risk model, to improve its implementation within LINK-UP, and then to expand the model into additional domains.

A summary of significant future work includes:

- Improved support for collaborative design teams, through risk management and time-based visualization, within LINK-UP
- Extension of the reuse paradigm to include process-related knowledge and the integration of both process and product-related risks
- Increased use of notification technology in alerting design teams to common problems through the reuse of risk-related knowledge

The initial comprehensive risk model presented in Chapter 4 was developed as a proof of concept in determining the benefit of risk management to the design of interactive systems, and the model proved sufficient for the purposes of answering the key research questions discussed in Chapter 5. Nevertheless, as research continues toward bridging the gap between HCI, software engineering, and project management, updated versions of the risk model will help to illustrate new ideas and answer additional questions. The knowledge gained through the development and evaluation of this initial model serves as a guide for future work.

Based on feedback acquired during the initial risk model evaluation, a second version of the model should be developed to improve usability and estimation accuracy. According to designers surveyed during the evaluation, increased accuracy relies on increasing the worth of stakeholder concern ratings and internal evaluation results; however, it is important to maintain additional factors so that the risk model can calculate a suitable estimation much earlier in the design process when information from stakeholders and evaluations is not available. To improve the risk model, additional user evaluations are needed to determine which factors contribute to the accuracy of both early and later risk estimations and which can be removed from the model without sacrificing the results of the prioritization.

Future versions of the model must also address the situation in which various degrees of risk-related information are available for different claims in a project set. Ideally, all information available for each individual claim should be used to calculate the most accurate estimation of risk for that claim. However, the inclusion and exclusion of various factors in the current risk equations will skew the prioritization results for a given set of claims. For example, if an empirical evaluation has been conducted on the downsides of one claim in a project set, but not on another, then the risk exposure values for those two claims as calculated by the current model will be difficult to compare. One possible solution to this problem is to calculate risk exposure based on the maximum number of risk factors available for *all* project claims. A more ideal solution is to redefine the risk equations to allow for comparison of claim risk exposures regardless of the amount of information available for each claim.

A number of the designers surveyed during the evaluation also commented on the prioritization of individual downside risks as opposed to claims. Although individual downsides represent the actual design risks, the initial risk model prioritizes claims both to increase the simplicity and scalability of the model and to preserve the tie between related downsides and the contextual information contained within a claim. Additional user evaluations are also

needed to determine the benefits and limitations of prioritizing downsides instead of clams. Subsequent versions of the risk model might consider prioritization of individual downsides, provided that an adequate link is preserved between each downside and its associated claim.

In addition to improving the definition of the risk model, future work can also serve to improve tool support for collaborative teams within LINK-UP. Although task allocation based on individual downside mitigation was not seen as a benefit by the majority of teams surveyed during the user evaluation, additional studies are needed to determine the benefit of mitigation task allocation on larger project teams working on larger, more complex design project. Additionally, the implementation of a risk-based timeline visualization would allow design teams to monitor the evolution of design risks throughout the course of a project.

The current risk model and the current structure of a claim are defined for the management of product-related design knowledge. To be a true benefit from a project management perspective, the model must be extended to encompass both product and process-related knowledge. The first step in this process is to define process-related claims. Unlike product-related design knowledge, rationale in support of process-related knowledge cannot always be extracted from literature. Thus, the concept of rationale, as well as critical parameters, claim quality, and other factors, must be reconsidered with respect to process-related issues. The interactions between process and product-related knowledge must also be defined. For example, do relationships exist between product and process-related claims? Can a process-related claim mitigate the downside of a product-related claim, and vice versa? How should product and process related claims interact within the context of a prioritized risk list? These questions must be answered before process-related knowledge can be successfully integrated within HCI design.

Finally, to support risk management in collaborative design through knowledge reuse, an increased use of notification technology could provide risk-related alerts to designers based on knowledge from previous projects. The information stored within a team memory throughout the course of a design project can be archived for reuse. If this team memory contains information about the project's design risks and mitigation solutions, then that knowledge can be leveraged to alert designers of potential problems at the start of similar, subsequent projects. As particular risks become more likely or potentially more costly throughout the course of the project, notification systems can also alert designers of critical changes in risk priority. Consequently, designers can learn of potential problems as early as possible and examine solutions from previous projects to guide their mitigation strategies.

Bibliography

[1] "The Delphi method: Techniques and applications", ed. H.A. Linstone and M. Turoff. 1975, Reading, Mass.: Addison-Wesley Publishing Co., Advance Book Program.

[2] "The American Heritage® Dictionary of the English Language: Fourth Edition". 2000.

[3] (SEI), CMU Software Engineering Institute, "Risk Management Overview ". 2005.

[4] Arias, Ernesto, Eden, Hal, Fischer, Gerhard, Gorman, Andrew, and Scharff, Eric, "Transcending the individual human mind: creating shared understanding through collaborative design". *ACM Transactions on Computer-Human Interaction (TOCHI)*, 2000. 7(1): p. 84 - 113.

[5] Aven, Terje, "Foundations of Risk Analysis: A Knowledge and Decision-Oriented Perspective". 2003: John Wiley & Sons.

[6] Barnum, C. M. "Building a Team for User-Centered Design". in *Proceedings of 2000 Joint IEEE International and 18th Annual Conference on Computer Documentation*. 2000.

[7] Basili, V. R. "The Experience Factory: packaging software experiences". 1989.

[8] Beise, Catherine M. "Employees and impact on work: IT Project Management and Virtual Teams". in *Proceedings of the 2004 SIGMIS conference on Computer personnel research: Careers, culture, and ethics in a networked environment*. 2004.

[9] Bernstein, Peter L., "Against the Gods: The Remarkable Story of Risk". 1996: Wiley.

[10] Boehm, Barry and Port, Daniel. "Educating Software Engineering Students to Manage Risk". in *Proceedings of the 23rd International Conference on Software Engineering*. 2001.

[11] Brown, Norm, "Industrial-Strength Management Strategies". *IEEE Software*, 1996. 13(4): p. 94 - 103.

[12] Carroll, J. M., "Making use: a design representation", in *Communications of the ACM*. 1994. p. 29-35.

[13] Carroll, J. M., "Making use: scenario-based design of human-computer interactions". 2000: The MIT Press.

[14] Carroll, John M. and Rosson, Mary Beth, "Getting around the task-artifact cycle: how to make claims and design by scenario", in *ACM Transactions on Information Systems*. 1992. p. 181-212.

[15] Carroll, John M., "Activity as the Object of Design", in *NSF Workshop - Science of Design: Software-Intensive Systems*. 2003.

[16] Carroll, John M., Neale, Dennis C., Isenhour, Philip L., Rosson, Mary Beth, and McCrickard, D. Scott, "Notification and Awareness: Synchronizing Task-Oriented Collaborative Activity". *International Journal of Human-Computer Studies*, 2003. 8(5).

[17] Charette, Robert N., "Software Engineering Risk Analysis and Management". 1989: Multiscience Press, Inc.

[18] Chewar, C. M., Bachetti, Edwin, McCrickard, D. Scott, and Booker, John. "Automating a Design Reuse Facility with Critical Parameters: Lessons Learned in Developing the LINK-UP System". in *Proceedings of the 2004 International Conference on Computer-Aided Design of User Interfaces (CADUI '04)*. 2004. Island of Madeira, Portugal.

[19] Cohen, Jacob, "Statistical Power Analysis for the Behavioral Sciences". 2nd ed. 1988: Lea.

[20] Cohen, S.G. and Bailey, D.E., "What Makes Teams Work: Group Effectiveness Research from the Shop Floor to the Executive Suite". *Journal of Management*, 1997. 23(3): p. 239-290.

[21] Davenport, Thomas H. and Prusak, Laurence, "Working knowledge: how organizations manage what they know". 1998: Harvard Business School Press.

[22] Decisioneering, Inc., Crystal Ball Risk Analysis Software and Solutions, "Financial Risk Analysis Example Models". 2005.

[23] Freeman, Peter and Hart, David, " A science of design for software-intensive systems ". *Communications of the ACM*, 2004. 47(8): p. 19 - 21.

[24] Fussell, Susan R., Kraut, Robert E., Lerch, F. Javier, Scherlis, William L., McNally, Matthew M., and Cadiz, Jonathan J. "Coordination, Overload and Team Performance: Effects of Team Communication Strategies". in *Proceedings of the 1998 ACM conference on Computer supported cooperative work*. 1998.

[25] Geyer, Werner, Richter, Heather, Fuchs, Ludwin, Frauenhofer, Tom, Daijavad, Shahrokh, and Poltrock, Steven. "A Team Collaboration Space Supporting Capture and Access of Virtual Meetings". in *Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work*. 2001.

[26] Grudin, Jonathan, "Groupware and social dynamics: eight challenges for developers ". *Communications of the ACM*, 1994. 37(1).

[27] Heemstra, Fred J., Kusters, Rob J., and Man, Huibert de. "Guidelines for Managing Bias in Project Risk Assessment". in *Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE 03)*. 2003.

[28] Heir, Manveer, Hoon, Harish, Terrell, Goldie, and McCrickard, D Scott, "Online Enlightenment: A Notification System for Online Status". 2004: Technical Report TR-04-30, Computer Science, Virginia Tech.

[29] Houdek, Frank and Kempter, Hubert. "Quality Patterns - An Approach to Packaging Software Engineering Experience". in *Proceedings of the 1997 symposium on Software reusability*. 1997.

[30] Jang, Chyng-Yang, Steinfield, Charles, Pfaff, Ben, "Supporting awareness among virtual teams in a web-based collaborative system: the teamSCOPE system". *ACM SIGGROUP Bulletin*, 2000. 21(3): p. 28 - 34.

[31] Jones, Capers, "Patterns of Software System Failure and Success". 1995: Intlernational Thomson Computer Press.

[32] Keshlaf, Ayad Ali and Hashim, Khairuddin. "A Model and Prototype Tool to Manage Software Risks". in *Proceedings of the First Asia-Pacific Conference on Quality Software*. 2000.

[33] Krueger, Charles W., "Software reuse". *ACM Computing Survey (CSUR)*, 1992. 24(2): p. 131-183.

[34] MacCrimmon, K.R. and Wehrung, D.A., "Taking Risks: the Management of Uncertainty". 1986, New York: The Free Press.

[35] MacQueen, Jason, "The structure of multifactor equity risk models". *Journal of Asset Management*, 2003. 3(4): p. 313 - 322.

[36] Malone, Thomas W. and Crowston, Kevin, "The interdisciplinary study of coordination", in *ACM Computing Surveys*. 1994. p. 88-119.

[37] McCrickard, D. Scott, Chewar, C. M., Somervell, Jacob P., and Ndiwalana, Ali, "A Model for Notification Systems Evaluation--Assessing User Goals for Multitasking Activity". *ACM Transactions on Computer-Human Interaction (TOCHI)*, 2003. 10(4): p. 312-338.

[38] Newman, W. "Better or Just Different? On the Benefits of Designing Interactive Systems in terms of Critical Parameters". in *Proceedings of the 1997 Conference on Designing Interactive Systems*. 1997: ACM.

[39] Norman, D. A., "Cognitive Engineering", in *User Centered Design: New Perpsectives on Human Computer Interaction*, D.A. Norman and S.W. Draper, Editors. 1986, Erlbaum: Hillsdale, NJ. p. 31-62.

[40] Norman, D. A., "Things That Make Us Smart: Defending Human Attributes in the Age of the Machine". 1993, Reading, MA: Addison-Wesley Longman Publ. Co., Inc.

[41] NSF, National Science Foundation, "Science of Design: Software-Intensive Systems". 2003.

[42] Palisade, Corporation, "@Risk Product Website". 2005.

[43] Paré, Guy and Dubé, Line. "Virtual teams: an exploratory study in key challenges and strategies". in *Proceeding of the 20th international conference on Information Systems*. 1999.

[44] Payne, Catherine, Allgood, C. F., Chewar, C. M., Holbrook, Chuck, and McCrickard, D. Scott. "Generalizing Interface Design Knowledge: Lessons Learned from Developing a Claims Library". in *Proceedings of the 2003 IEEE International Conference on Information Reuse and Integration*. 2003.

[45] Powell, Anne, Piccoli, Gabriele, and Ives, Blake, "Virtual teams: a review of current literature and directions for future research". *The DATA BASE for Advances in Information Systems*, 2004. 35(1): p. 6-36.

[46] Rosson, M. B. and Carroll, J. M., "Usability Engineering: Scenario-Based Development of Human-Computer Interaction." 2002, New York, NY: Morgan Kaufman.

[47] Rowe, William D., "An Anatomy of Risk". 1988, Malabar, FL: Robert E. Krieger Publishing Co.

[48] Roy, Geoffrey G. "A risk management framework for software engineering practice". in *Proceedings of the 2004 Australian Software Engineering Conference*. 2004.

[49] Royce, Walker, "Software project management: a unified approach". 1998. Addison-Wesley.

[50] Simon, Herbert, "The Sciences of the Artificial, 3rd Edition". 3rd ed. 1996, Cambridge, MA: MIT Press.

[51] Smith, Jamie L., Bohner, Shawn A., and McCrickard, D. Scott. "Project Management for the 21st Century: Supporting Collaborative Design through Risk Analysis". in *Proceedings of the ACM Southeast Conference (ACMSE '05)*. 2005.

[52] Standish Group, The, "CHAOS Chronicles". 2005.

[53] Steinfield, Charles, Jang, Chyng-Yang, and Pfaff, Ben. "Supporting virtual team collaboration: the TeamSCOPE system". in *Proceedings of the international ACM SIGGROUP conference on Supporting group work*. 1999.

[54] Sutcliffe, Alistair, "On the effective use and reuse of HCI knowledge". *ACM Transactions on Computer-Human Interaction*, 2000. 7(2): p. 197-221.

[55] Sutcliffe, Alistair, "The Domain Theory: Patterns for Knowledge and Software Reuse". 2002: Erlbaum Assoc.

[56] Sutcliffe, Alistair G. and Carroll, John M., "Designing claims for reuse in interactive systems design", in *International Journal of Human-Computer Studies*. 1999. p. 213-242.

[57] Wahid, Shahtab, Allgood, C. F., Chewar, C. M., and McCrickard, D. Scott. "Entering the Heart of Design: Relationships for Tracing Claim Evolution". in *Proceedings of the Sixteenth International Conference on Software Engineering and Knowledge Engineering (SEKE '04)*. 2004.

[58] Wallace, Linda, Keil, Mark, and Rai, Arun, "How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model". *Decision Sciences*, 2004. 35(2): p. 289 - 321.

[59] Williams, Ray C., Walker, Julie A., and Dorofee, Audrey J., "Putting risk management into practice". *IEEE Software*, 1997. 14(3): p. 75 - 82.

[60] Zhang, Jeff, Zage, Dolores, and Zage, Wayne. "Improving project planning/tracking for student software engineering projects through SOPPTS". in *Proceedings of the 16th IEEE Conference on Software Engineering Education and Training*. 2003.

Appendix A : Online Enlightenment (OE) Case Study

The Online Enlightenment project created an off-the-desktop notification system to alert users of an HCI lab as to which of their lab partners are currently available through an online instant messaging system. Since instant messaging has become a staple in the professional world for use in coordinating project tasks, meetings, and deadlines, regular lab users, as well as visitors, can benefit from the ability to see who is available without logging into the messaging system on an individual computer.

OE is a physical device that presents online presence information in a semi-public space. The device uses a map metaphor to represent the layout of a lab, showing online instant messenger status for members of the community. Users of the device can combine information from the device with information from their lab physical environment to identify unfamiliar lab members, determine human-to-human interaction strategies, and plan meetings.

Development of the OE interface was inspired by lab users' desire to locate other members of the lab, as well as the lab director's desire to keep track of his research students. Additionally, the system was developed as a physical example of a notification system, allowing lab visitors to not only locate members of the lab, but also understand the research taking place within the lab. The design of OE followed a claims-centric, scenario-based approach. A sample problem scenario, depicting the difficulty that regular lab users experience in locating their lab mates, is presented below.

Lab Users Scenario:

Jason is working at his computer in the HCI lab. He asks Ali a question and is told that Christa would have the necessary information. Jason checks his MSN buddy list to find that Christa is not currently online. He configures MSN to sound an audible alert when Christa signs into the system, then minimizes his buddy list window and continues working. Unfortunately, if the tone sounds while Jason is out of the room, he will not hear it. When he returns to the lab, the only way to find out if Christa has signed into MSN is to restore his buddy list window and check her status.

Jason continues working at his computer, leaving the lab several times to run errands in other parts of the building. He gets his lunch and then returns to his desk to continue working. After a while, Jason checks his MSN buddy list to find that Christa has been signed on for almost an hour. The tone must have sounded while he was out of the room.

Problem Claims:

Notification appears on computer screen	**IRC:** 1.0, 0.8, 0.0
+	interruption is obvious, if working at the computer
+	reaction is easy since you are already at the computer
+	notification using text provides a lot of information
+	use of keyboard and mouse well-known
-	takes up screen space
-	must be at computer, or notification must be persistent

Finding all the lab users using MSN	**IRC:** 1.0, 0.8, 0.1
+	easy to get all the information you need at once
+	intuitive to respond if needed
-	requires access to a computer
-	becomes the user's primary task
-	must continually check the list to monitor status of lab users
-	no history of information regarding the state of the user over a period of time
-	no information of how long the particular user has been in that state for

Lack of a dynamic interactive representation of lab status	**IRC:** -, -, 0.1
+	decreases clutter that can interrupt people in the lab from their primary tasks
+	desk and wall space can be dedicated to other (less dynamic but more information-rich) displays, such as research posters
+	discourages outsides from entering lab
-	provides no sense of direction for a visitor
-	can lead to a bland and boring lab area
-	visitors leave without a strong understanding of research activity in the lab
-	does not highlight the goals of HCI and notification systems research

Root concept:

A centrally located display providing current and historical information about lab users could improve the productivity of the lab by supporting lab users in locating and communicating with one another. The system must promote easy reaction to the information made available as well as minimal diversion from users' primary tasks. Minimizing attention diversion is important because many regular lab users would experience a decrease in productivity if the notification system constantly interrupted their normal tasks. By making status information easy to understand, the amount of time spent interfacing with the system would be minimal, allowing users to perform their normal tasks.

Primary Goals:	
Provide easy reaction to information regarding the availability of lab users	
Minimize interruption to users' primary tasks	
Secondary Goals:	
Tie the system to an existing instant messenger	
Attract and impress lab visitors	
Target System IRC:	(0.0, 0.6, 1.0)

The design proceeded through activity, information, and interaction design. A subset of the design claim set is presented below.

Highlighting design risks:

The goal for a redesign of the OE system is to mitigate the most critical downsides while maintaining or strengthening the most important upsides. Given the complexity of design, designers cannot expect to mitigate all possible downsides. Instead, they must prioritize those downsides and focus on mitigating the most important design issues. Our risk-driven management model prioritizes claims as design risks by estimating a risk exposure value for each claim. Risky claims are considered those with the highest chance of interfering with (as opposed to contributing to) the design goals for the system. Consequently, the critical downsides in the high priority claims should be addressed first during redesign.

Several important numerical values are provided with each of the eight claims presented below: an IRC rating for the claim, a claim quality rating, a stakeholder concern rating for each downside, an internal evaluation factor for each downside, and a set of quality and relevance factors for each external source of rationale cited for a downside. These are the key factors used in determining claim priority.

The **claim IRC rating** is compared to the design model IRC rating for the overall system to indicate how well the claim might contribute to the project's design goals.

The **claim quality rating** is measured on a scale from 1.0 to 4.0, with 1.0 representing a low quality claim and a 4.0 representing a high quality claim. Claim quality considers factors such as the experience of the claim author, the quality of external sources of rationale supporting the claim, the type of artifact associated with the claim, and the degree to which the claim has been reused.

The **external rationale factor** accounts for both the quality of sources of rationale as well as their relevance to the context of the current design project. The quality factor considers the venue in which the work was published along with the type of results (analytic, empirical) that were presented in the publication. Relevance considers three key factors in notification systems design –the accessibility of the system (public, semi-public, private), the size of the display (large-scale, desktop-scale, handheld), and the use of the display (primary, secondary).

The **stakeholder concern rating** (*stake*) measures the degree to which project stakeholders are willing to accept the effect of a particular downside. A value has been assigned to each downside along a 1.0-4.0 decimal scale, based on the following point indicators:

1.0 The effect of this downside will not interfere with my goals, mitigate opportunistically
2.0 Maintaining the upsides of this claim is more important than mitigating this downside, mitigate only if it is convenient to do so
3.0 Mitigation of this downside is important and should be given priority
4.0 Mitigation or elimination of this downside is absolutely essential to the success of the interface

The **internal evaluation effect factor** (*effect*) for the results of an analytic evaluation has been assigned to each downside by examining the interface and determining the effect of each particular downside. A value has been assigned along a 0.0 – 1.0 decimal scale, based on the following point indicators:

0.0 The effect of this downside is not an annoyance, nor is it interfering with user goal(s)
0.33 The effect of this downside is an annoyance but does not make user goal(s) more difficult to accomplish
0.67 The effect of this downside makes user goal(s) significantly more difficult to accomplish
1.0 The effect of this downside makes user goal(s) impossible to accomplish

These values contribute to the downside risk exposure value (DSE) calculated for each downside. The maximum DSE per claim (highlighted in bold) then contributes to the overall risk exposure value for that claim. Although high priority claims do not always indicate the existence of a critical design problem, they do alert the designer to possible risks that deserve consideration during redesign.

OE system characteristics:

Design model IRC: (0.0, 0.6, 1.0)

Design size: (2) Desktop-scale
Design accessibility: (2) Semi-private
Display use: (2) Secondary

Characteristics common to all 8 design claims:

Author experience: (2) Novice designers created these claims
Artifact quality: (2) As OE was being designed, simple screenshot prototypes existed
Degree of reuse: (1) All new claims, nothing has been reused
Average user rating: (1) None of the claims have been rated

These characteristics are accurate for the claims used in the evaluation projects as well.

Design Claims:

Activity claims:

Utilizing MSN Messenger to track online activity (claim 1)							**IRC:** 0.0, 0.8, 0.1			
Claim Quality: 1.5		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	easy to collect necessary information	3	2	x	x	x	x	x	x	x
+	provides information about an individual, not a computer	3	2	x	x	x	x	x	x	x

-	requires access to a computer	3	2	2.0	0.1	2	2	2	1.0	0.41
-	cannot distinguish users in the lab from remote users	3	2	2.5	0.4	2	2	2	1.0	1.02
-	users might forget to log into MSN	3	2	1.5	0.2	1	2	1	1.0	0.48
-	no history of information about the state of the user over time	3	2	3.0	0.6	2	2	2	1.0	**1.62**

Centrally located physical device (claim 2)							**IRC:** 0.0, 0.7, 0.3			
Claim Quality: 1.45		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	use of the display is moderately intuitive	2	2	x	x	x	x	x	x	X
+	does not require access to a computer	3	2	x	x	x	x	x	x	X
+	secondary display does not interfere with a user's primary task	3	2	x	x	x	x	x	x	X
+	a tangible display is aesthetically pleasing to lab visitors	2	2	x	x	x	x	x	x	X
-	use of the display is less flexible	3	2	1.5	0.1	2	2	2	1.0	0.33
-	text display area is limited	3	2	2.5	0.3	2	2	2	1.0	0.82
-	use of metaphors with a tangible display could be confusing to users	3	2	2.3	0.4	2	2	2	1.0	0.96
-	use of the display in a semi-public area raises privacy issues	2	1	2.8	0.4	2	1	2	1.0	**1.25**

Information/Interaction claims:

Use of a solid light to represent online status, a blinking light represents away status, and no light when a person is offline (claim 3)							**IRC:** 0.5, 0.7, 0.2			
Claim Quality: 1.55		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	easy to distinguish differences in lights	4	3	x	x	x	**x**	x	X	X
+	meaning of the lights is intuitive if a user is aware of the connection to MSN	2	2	x	x	x	x	x	X	X
-	does not make explicit the connection to MSN	2	2	2.0	0.4	3	2	1	1.0	0.88
-	blinking lights might be distracting to users working near the display	4	3	3.0	0.8	2	1	2	1.0	**2.07**

Use of pushbuttons for each person to change LCD information (claim 4)							**IRC:** 0.0, 0.5, 0.3			
Claim Quality: 1.5		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	users do not have to wait for information to be displayed	3	2	x	x	x	x	x	x	x
-	requires more space on the display	3	2	1.9	0.1	2	2	2	1.0	0.39

-	requires an intuitive layout of buttons on the display	3	2	3.5	0.3	2	2	2	1.0	**1.19**

Displaying caricatures of each person on a pushbutton (claim 5)							**IRC:** 0.0, 0.5, 0.7			
Claim Quality: 1.45		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	caricatures reduce the privacy concern since they are not as recognizable as photos	2	1	x	x	x	x	x	X	X
-	not obvious that the caricature is a pushbutton	3	3	3.2	0.6	2	2	2	1.0	**1.73**
-	some users might not recognize the caricature	2	2	2.7	0.4	2	2	2	1.0	1.10
-	wear on the buttons shows which users are most popular	3	2	2.3	0.3	2	2	2	1.0	0.78

LCD displaying textual information (claim 6)							**IRC:** 0.0, 0.4, 0.6			
Claim Quality: 1.475		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	increases comprehension by providing information that would be difficult to convey otherwise	3	2	x	x	x	x	x	X	x
+	results in a flashier display	1	1	x	x	x	x	x	X	x
-	displaying textual information might violate privacy	3	3	2.7	0.2	1	1	2	1.0	0.80
-	limited space on the LCD to display comprehendible messages	3	3	3.3	0.4	2	2	2	1.0	**1.27**

Display depicting a scaled map representation of the lab space (claim 7)							**IRC:** 0.0, 0.2, 0.6			
Claim Quality: 1.55		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	easy to understand the layout of pushbuttons on the display based on person's location in the lab	2	3	x	x	x	x	x	X	x
+	helps users to locate people they do not know or recognize	3	3	x	x	x	x	x	X	x
-	some areas of the display might become over-crowded while others are blank, due to the layout of people in the lab	3	2	3.1	0.1	2	2	2	1.0	0.61
-	cannot locate names alphabetically	3	3	2.4	0.2	1	2	1	1.0	**0.74**

Use of a wall-mounted display (claim 8)							**IRC:** 0.0, 0.2, 0.7			
Claim Quality: 1.5		*venue*	*results*	*stake*	*effect*	*acc*	*size*	*use*	*mit*	**DSE**
+	does not take up desk space	3	2	x	x	x	x	x	x	x
-	might be less noticeable to users	3	2	1.5	0.1	2	1	2	1.0	**0.35**

Priority list: 1

Priority	Risk Exposure	Claim feature
1	2.86	Use of a solid light to represent online status, a blinking light represents away status, and no light (claim 3)
2	2.41	Utilizing MSN Messenger to track online activity (claim 1)
3	2.27	Displaying caricatures of each person on a pushbutton (claim 5)
4	1.97	Centrally located physical device (claim 2)
5	1.91	Use of pushbuttons for each person to change LCD information (claim 4)
6	1.88	LCD displaying textual information (claim 6)
7	1.38	Display depicting a scaled map representation of the lab space (claim 7)
8	0.98	Use of a wall-mounted display (claim 8)

Priority list: 2

Priority	Risk Exposure	Claim feature
1	3.0	LCD displaying textual information (claim 6)
2	2.75	Display depicting a scaled map representation of the lab space (claim 7)
3	2.73	Displaying caricatures of each person on a pushbutton (claim 5)
4	2.70	Use of pushbuttons for each person to change LCD information (claim 4)
5	2.50	Use of a solid light to represent online status, a blinking light represents away status, and no light when a person is offline (claim 3)
6	2.275	Centrally located physical device (claim 2)
7	2.25	Utilizing MSN Messenger to track online activity (claim 1)
8	1.50	Use of a wall-mounted display (claim 8)

Appendix B : CS 3724 Semester Design Project: Phase 4-A

Design Risk Analysis

During this phase of the project, you will work as a group to begin the redesign of your project interface. Your goal for redesign is to improve your interface by mitigating a number of downsides, which represent design risks in your project. You are not expected to mitigate all of the downsides in your design. To improve your interface, you must identify the most critical design risks and focus on mitigating those.

Part 1: **Input data into LINK-UP**
First, input into LINK-UP the data you collected on paper during project Phase 2, Part 2 (Claim quality and risk analysis). LINK-UP will use this information to produce a prioritized list of your project claims. Follow the instructions below to input the necessary information into LINK-UP:

1. Log into your LINK-UP project account
2. Use the Claim Edit feature to edit each of your project claims, inputting the following information from your Phase 2 report (2.2: Categorizing sources of rationale):
3. Venue and Results values for each upside and downside of each claim
4. System Accessibility, Display Size, and Display Use values for each downside of each claim
5. Save your updates to each claim
6. Enter the Risk Analysis module
7. Click 'Continue to System Characteristics'
8. Input the three system characteristics (System Accessibility, Display Size, and Display Use) that you assigned during Phase 2 (2.1: Assigning System Characteristics)
9. Click 'Continue to Stakeholder Concern Ratings'
10. For each claim in your design set, input the stakeholder concern rating that you assigned to each individual downside during Phase 2 (2.3: Group Stakeholder Concern Ratings)
11. Click 'Continue to Analytic Evaluation Input'
12. Input the LINK-UP login name for the group that evaluated your system
13. For each claim in your design set, input the numerical Downside Effect Rating that your evaluating group assigned to each downside during the Downside Effect Activity in Phase 2.
14. Click on 'View Risk Priorities' to view your claim priority list

Part 2: Prioritization of project claims and selection of critical downsides
After you have input the necessary information into LINK-UP, click on the Risk Priorities tab in the Risk Analysis module to access a prioritized list of the claims included in the initial design of your project interface. Prioritization is based on a risk exposure value assigned to each claim. High priority claims have high risk exposure values and thus, have a greater chance of interfering with your design goals.

Discuss your prioritized claim list as a group and complete the chart provided below for each claim in your prioritized list. Complete one chart for each project claim.

Claim ID: _____ Claim feature: _____

Original Priority	Updated Priority

Characteristic(s) supporting priority	Explanation

Downside(s) you plan to mitigate	Characteristic(s) prompting mitigation

First, record the priority of the claim in the original list, and then update the priority of each claim as you see fit. A number of factors contribute to the risk exposure value of a claim, including characteristics of the claim as a whole and characteristics of the claim's individual downsides. Base your decision to re-prioritize a claim on one or more of the claim and downside characteristics described below. List the characteristics that affected your decision in the column of the chart labeled "Characteristic(s) supporting priority."

1. **Claim quality:** the quality of a particular claim as a whole, which takes into account the experience of the claim author, the quality of external sources of rationale, the quality of the any artifact associated with the claim, the degree to which the claim has been reused, and an average user rating for the claim
2. **IRC difference:** the degree of difference between the claim IRC rating and the design model IRC rating
3. **Stakeholder concern ratings**: the degree to which project stakeholders are willing to accept the effect of a particular downside in the design of the system
4. **External rationale:** the degree to which external sources of rationale supporting a particular downside present evidence of its effect in similar systems
5. **Internal evaluation results:** the degree to which the results of an evaluation of the interface show evidence that a particular downside is effecting the interface
6. **Other:** (please specify)

In the "Explanation" column, explain why you based your decision on certain characteristics. For example:

- If the **Claim quality** characteristic affected your decision to lower the priority of a particular claim, you might elaborate by stating that…
  - *The quality of this claim is very low and I don't see where this characteristic factors into the prioritization*
- If the **Stakeholder concern rating** characteristic affected your decision to raise the priority of a claim, you might elaborate by stating that…
  - *Project stakeholders are very concerned about a particular downside in this claim, but the claim seems too low on the list*
- If the **Internal evaluation results** characteristic affected your decision not to change the priority of a claim, you might elaborate by stating that…
  - *Results of the analytic evaluation show that none of the downsides in this claim are a problem, and the priority seems to accurately reflect those results*

You may choose to maintain the priority of some or all of the claims in your list. You must provide one or more characteristics and an explanation for every claim, even if the priority remains the same.

Although the factors described above do not always determine the definitive existence of a critical design risk, they do flag potential problems that warrant further examination. The high-priority downsides within a high-priority claim should be considered for mitigation during the redesign phase of your project. As you consider the priority of a claim, determine which downsides in that claim can and should be mitigated during the redesign phase of your project. For each claim, list the downside(s) that you plan to mitigate in the appropriate column in the chart.

In the column labeled "Characteristic(s) prompting mitigation," provide one or more of the following reasons for why you chose to mitigate each downside:

1. **Mitigation is essential:** Mitigation of this downside is critical to the success of the interface
2. **Mitigation is straightforward:** Mitigation of this downside will require minimal time and effort
3. **Mitigation to improve interface**: Mitigation of this downside is not essential to the success of the interface; however, mitigation will significantly improve the system and is worth the effort
4. **Mitigation to maintain upsides:** The upsides of the claim associated with this downside are important to maintain; thus, this downside must be mitigated to ensure claim effectiveness in the design
5. **Other:** (please specify)

You may choose to mitigate zero, one, or more than one downside from each claim, depending on the importance of each downside to the success of your interface. Remember that you will have approximately one week to redesign your interface. Keep this time frame and other project constraints in mind as you determine which downsides to mitigate. The goal is not to mitigate the most downsides, but to choose a small subset of the most important downsides to mitigate. You are expected to choose approximately 3-5 downsides that you plan to mitigate.

Part 3: <u>Follow-up questions</u>
Discuss and answer each of the following questions as a group. Consider the work you have done during this phase of the project and how it has helped in planning the redesign phase of your semester project. Clearly label your answers in your Phase 4-A report.

1. How beneficial is prioritization of project claims in terms of guiding design improvements?
   - O Extremely beneficial
   - O Somewhat beneficial
   - O Neutral
   - O Somewhat unbeneficial
   - O Extremely unbeneficial

   Comments:

2. How beneficial would the prioritization of individual downsides be in terms of guiding design improvements?
   - O Extremely beneficial
   - O Somewhat beneficial
   - O Neutral

- O Somewhat unbeneficial
- O Extremely unbeneficial

Comments:

3. How important is each of the claim and downside characteristics that contribute to its risk exposure value? Rank the following characteristics in order of importance, with 1 being the most important for estimating the risk exposure value of a claim:

   Rank

   ____ Stakeholder concern for the downsides of the claim
   ____ Quality of the claim
   ____ Difference in claim and design model IRC
   ____ External sources of rationale supporting each tradeoff
   ____ Results of internal evaluation of the interface
   ____ Other (explain)

4. Do you plan to approach the redesign phase of the project by assigning responsibility for individual downside mitigation to specific team members?

   - O Yes
   - O No

   Why or why not?

Summary of Phase 4-A deliverables:

Produce a Design Risk Analysis report including the following information:

1. Your original priority list from LINK-UP
2. A completed risk priority chart for each claim in your project set
3. Responses to the four follow-up questions in Part 3

Submit your Phase 4-A report to the Blackboard Digital Drop box by Friday, April 1 at 6:00PM.

Appendix C : CS 3724 Semester Design Project: Phase 4-B

Reflections on System Redesign

Part 1: Record of downside mitigation
Use the chart below to record the specific downsides that you mitigated during the redesign of your system. **Complete one chart for each claim in which one or more downsides were mitigated during the redesign of your system**. List the ID, feature, and original priority of the claim, along with each downside that was mitigated and the ID and feature of the mitigating claim.

Claim ID: _____ Original Priority: _____

Claim feature: _____

Downside(s) mitigated	Claim mitigating this downside	
	ID	**Feature**

How does mitigation of this downside(s) improve your interface?

You were expected to mitigate a small subset of possible design risks, which means a number of downsides from high priority claims were left unresolved during this design iteration. Consider some of the downsides in your project that you would have liked to mitigate but chose to ignore during this redesign phase.

Why did you choose to ignore these high priority downsides? Select all reasons that apply, and list one or more downsides that were ignored for each reason you select. **Particularly, mention any downsides that you planned to mitigate during Phase 4-A but chose not to mitigate during redesign.**

☐ Mitigation was not as critical as depicted by the priority list

Claim ID	**Downside**
Comments	

☐ Mitigation introduced new downsides that were critical to avoid

Claim ID	**Downside**

Comments	

☐ Mitigation of this downside was too time-consuming

Claim ID	**Downside**
Comments	

☐ Mitigation was not feasible for reasons other than time (please explain)

Claim ID	**Downside**
Comments	

Part 2: <u>Follow-up questions</u>
Discuss and answer each of the following questions as a group. Consider the work you have done during phases 4-A and 4-B of the project. Clearly label your answers in your Phase 4-B report.

1. To what extent did you follow the prioritized claim list during the redesign of your system?
   - O We did not consider claim priority at all when determining which downsides to mitigate
   - O We considered claim priority during redesign, but the ultimate decision to mitigate a downside was made regardless of claim priority
   - O We referred to the priority list throughout redesign and mitigated downsides from high priority claims whenever possible
   - O We followed the list closely and refused to mitigate downsides from claims that were not in the top one-third of the priority list

   Comments:

2. How useful was the priority list in guiding your redesign?
   - O Of no use
   - O Somewhat useless
   - O Neutral

- o Somewhat useful
- o Extremely useful

Comments: