Entangled Design Knowledge: Relationships as an Approach to Claims Reuse

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

Virginia Polytechnic Institute and State University, United States Military Academy

FOUR RELATIONSHIP APPLICATIONS FOR REUSE

Center for Human-Computer Interaction[1]
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061-0106
{swahid, mccricks, chonglee}@cs.vt.edu

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

Shahtab Wahid is a computer scientist with an interest in HCI design knowledge reuse; he is a Ph.D. candidate in the Department of Computer Science at Virginia Tech, Blacksburg, VA, USA. **D. Scott McCrickard** is a computer scientist with interests in notification systems, methods in HCI design, and reuse; he is an Assistant Professor in the Department of Computer Science and is a member of the Center for HCI at Virginia Tech, Blacksburg, VA, USA. **C. M. Chewar** is a computer scientist with an interest in notification systems, methods in HCI design, and reuse; she is an Instructor in the Department of Electrical Engineering and Computer Science at the United States Military Academy, West Point, NY, USA. **Jason Chong Lee** is a computer scientist with interests in usability and software engineering methodologies and design rationale; he is a Ph.D. candidate in the Department of Computer Science at Virginia Tech, Blacksburg, VA, USA.

ABSTRACT

As a discipline, human-computer interaction produces creative and innovative designs that could provide a reusable collection of design knowledge on which future efforts could build. It is unfortunate that so much of this knowledge is not fully reused by designers today. To encourage the use of previously identified HCI knowledge, we propose a model of reuse building on Carroll's notion of claims, design knowledge components that capture the positive and negative psychological effects of design features. We address four challenges associated with reuse in a library of claims, adopted from software engineering—a discipline in which the notion of reuse has been prevalent for quite some time. Building on Krueger's definition of reuse and his conceptualization of four key aspects—abstraction, selection, specification, and integration—we propose a reuse approach based on incorporating these four aspects into the design process. To abstract, select, specify and integrate claims, we identify claim relationships, descriptions of connections between claims. We portray how claim relationships can be used to aid in identifying claim types, searching for claims, creating new claims, and aggregating claims. By integrating relationships into a claims library, we demonstrate how they can be applied to assist claims reuse and present studies related to each application of the relationships.

CONTENTS

1. INTRODUCTION

Good practitioners of human-computer interaction (HCI) generally produce creative and innovative designs, but often ignored are incremental design approaches that combine and build upon established techniques and approaches. Our research approach seeks ways to capture the collective design knowledge of the HCI community and appropriately deliver pieces of this knowledge to designers. For example, an important finding of a study can be encapsulated in the form of knowledge and then shown to designers so that they can account for the finding in their own work. The concept of knowledge transfer is imperative since it can often be harder to create new design knowledge from previous research than to reuse. Therefore, we must strive to understand how the reuse of this knowledge can be furthered.

Systems based upon design knowledge that has previously been identified and erected using fundamental theories and observational studies show promise in achieving higher quality designs. Hence, there is a developing need to reuse design knowledge during the development process. Such a development method can potentially ease and decrease overheads such as time and cost in design. We turn to the software engineering community where the idea of reuse has been more prevalent (Dusink & van Katwijk, 1995)(Gall, Jazayeri, & Klosch, 1995)(Krueger, 1992). For example, the reuse of design patterns was put to the forefront in software engineering by Gamma, Helm, Johnson, and Vlissides (1995), the Gang of Four. Although patterns now have a dedicated community that has long been exploring their reuse in software engineering, their use is also explored in HCI (Borchers, 2000). This research should inspire us to consider how other forms of knowledge can be reused to benefit HCI.

The benefits of reuse for HCI design knowledge have previously been explored (Sutcliffe, 2000)(Sutcliffe & Carroll, 1999). Instead of asking designers to create their own design from scratch and yield radical designs each time, we would like to see designers use previously identified design knowledge during their design process, giving them the ability to consider previous research and design efforts. Whittaker, Terveen, and Nardi (2000) argue radical invention is vital to making progress, but that designers should try to always make improvements based on prior work. It is argued that only when designers refer to prior work and can no longer improve upon it does it make more sense to consult radical invention.

Practitioners reusing design knowledge will be forced to consider the effects of certain features, giving them the opportunity to incrementally improve upon previous work. As a byproduct of this argument, we see the creation of knowledge as another equally important key factor. Through observational and theoretical backing, designers should understand how to isolate a certain feature and express its effects. This knowledge should then be written in a form such that it can be reused by future designers. It is our objective to explore how we can initiate the reuse of HCI design knowledge.

Many schemes for reuse incorporate the use of a repository containing reusable components and methods for searching for components (e.g., (Payne et al.,

2003)(Henninger, 1997)). However, many of these repositories lack specific design features that further reuse. Reusable components have to be described by descriptions of types to avoid having designers peruse through the details of each component. When looking for what one needs, designers must be able find the most appropriate component while preserving the context of their search and knowing what is available to them. It is very likely designers may not find what they want and therefore need a new component based off of components that already exist to suit their own needs. Finally, there must be a method to connect all of the pieces together to form a whole design. To respond to the absence of the features mentioned above, we seek a way of adopting a software engineering approach to reuse and applying it to HCI. We introduce our own digital library, a collaborative *design knowledge repository*, and show how we can design it to mitigate the problems of reuse.

The software engineering discipline has traditionally researched the area of reuse, providing many examples of early research (Krueger, 1992)(Biggerstaff & Perlis, 1989a)(Biggerstaff & Perlis, 1989b). Krueger (1992) describes the most innate characteristics of reuse and provides a comprehensive overview of reuse techniques. His perspective on reuse has become a well-recognized piece of research. He advocates every technique to reuse should be based on four key aspects: *abstraction, selection, specification,* and *integration*. These aspects describe the processes of describing, searching, modifying, and combining within reuse. Our belief is these processes are so inherent to reuse, our repository must be designed to specifically support these aspects to maximize the amount of reuse. We wish to take these aspects and establish a cycle of reuse by creating methods for designers to extract design knowledge from the repository and lead to future contributions to the repository.

Our reuse technique is grounded in *design knowledge relationships*, succinct descriptions of connections between pieces of design knowledge. We propose the use of design knowledge relationships and explain their role in solving the guiding problem of reuse. But how exactly does one abstract, select, specify, and integrate using relationships? The proposed relationships can be used to describe, identify, search, create, and aggregate design knowledge—concerns critical to the design of our digital library. Guided by identified problems we aim to solve, we portray how design knowledge relationships can be applied to explicitly support each aspect of reuse. Through separate studies, we provide a proof of concept for why design knowledge relationships should be used to facilitate reuse. This effort strengthens the process of how systems are created by reusing design knowledge and demonstrates why the use of design knowledge relationships is so important within this process.

We elaborate on our work in Section 2 by first discussing reuse, design knowledge, the design knowledge repository we use, and relationships. Section 3 establishes Krueger's vision of an approach to reuse, defines the four reuse aspects, and maps the aspects to problems associated with our own repository. The complete definition of our approach to reuse using relationships is exposed in Section 4. Section 5 illustrates how our design knowledge relationships can be applied in four different ways to exhibit characteristics of the reuse aspects and provide solutions to the identified problems. Finally, we provide closing thoughts and future work in Section 6.

2. BACKGROUND

In this section we cover the topics upon which this research is based. We first explain the motivation to reuse and what is needed for it. This is used as motivation for the reuse of *claims*, the design knowledge components concerning our work. Following this explanation, we introduce our own *claims library*, a design knowledge repository for reuse. Finally, an overview of the concept of relationships and their usefulness in linking claims and aiding reuse is discussed.

2.1. Reuse

Reuse, the idea of activities reusing previously created artifacts consisting of pieces of formalized components, has attracted a lot of research. A collection of research on reuse was compiled by Biggerstaff and Perlis (1989a & 1989b), representing early advances in models and applications of reuse by the software engineering community. The general claim in reuse, if applied properly, is that it can speed up the development process by reducing the amount of time and effort put into previously solved issues (Dusink & van Katwijk, 1995). This eliminates the need to rethink problems that have already been solved by others, providing incentive for the reuse of knowledge in the field of HCI.

Reusable components must be found, selected, understood, and, if needed, adapted (Dusink & van Katwijk, 1995) by designers during the design process. These components are typically retrieved from a reuse repository. The success of a user locating a potentially useful component can depend on several factors, including their familiarity with the repository and degree to which characteristics of a component have been specified (Creech, Freeze, & Griss, 1991). Gall, Jazayeri, and Klosch (1995) provide directions for future reuse research. They maintain research in reuse should focus on creating a component-based industry, providing impetus behind our argument for designing for reuse.

Artifacts must be created and made available for future reuse, but this is not an easy task. In many cases we see artifacts that are created during a design process and then permanently set aside, hindering efforts to mold the artifacts into reusable components. Designers prefer not to think about making certain components abstract enough such that they can be reused by others in the future, drenching hopes of designing for reuse. Even if artifacts are created, an equally puzzling concern is making them available to other designers.

Borchers (2000) argues for the use of design patterns, another form of design knowledge, to capture HCI knowledge. He mentions the need for the encapsulation of the designers' experiences, methods, and values into patterns. Landay and Borriello (2003) created patterns for ubiquitous computing. Their goal is to apply them within a field by documenting lessons learned and passing them on to new designs. Such research efforts into the reuse of patterns provide impetus behind the argument to consider other forms of reusable design knowledge to benefit HCI.

2.2. Design Knowledge: Claims

Scenario-Based Design (SBD) (Rosson & Carroll, 2002) is a design process in which scenarios, narratives describing a particular task, are used as a base for creating interactive designs. SBD uses various types of scenarios to guide the design process. An analytic evaluation process within SBD, called claims analysis, identifies scenario features that have usability consequences and stores this information in a structure called a *claim*. Carroll's claims, originating from the Task-Artifact Theory (Carroll, Kellogg, & Rosson, 1991), are design knowledge components which capture the positive and negative effects of an artifact within a usage context (Carroll, 1994)(Carroll and Kellogg, 1989)(Carroll, Singley, & Rosson, 1992). Delivered in informal natural language, claims address a variety of situational and interface aspects that affect the compatibility of the design and user models, such as user satisfaction and feeling of reward, color and object layout, and strength of affordances. Inherently objective, claims provide designers with a pure view into what makes an artifact live and breathe, grounded in theories and observations of user experiences.

Carroll introduced claims as a way to capture design knowledge. Generally, the claim concept has been used as a disposable knowledge unit to guide conceptualization of a design. Sutcliffe proposed and spearheaded efforts to make claims into reusable design knowledge components, lasting well beyond the designs they were initially created for. An extensive structure for claims and the idea of storing claims in a library was presented (Sutcliffe & Carroll, 1999)(Sutcliffe, 2000). To demonstrate the model of a claim, we present one about the collage metaphor and assess its effects (see Figure 1). A collage metaphor stems from the notion that artifacts are placed haphazardly in an unorganized fashion, much like a public bulletin board (Greenberg & Rounding, 2001).

Organizing information items using a collage metaphor
+ Allows users to informally post information without any regard to organization
+ Allows users to gain an understanding of an item's age/applicability with respect to the number of items that may be covering it
+ Lack of information categorization accommodates a wide range of different types of information to be placed
- BUT the lack of organization can hinder efforts to find a particular information item
- BUT overlapping items may force users to move items in order to fully reveal themselves

Figure 1. An example of a claim showing a title, upsides, and downsides.

One can imagine a situation where a designer is trying to apply an organizational technique to a public display. The construct of the claim in Figure 1 would allow a designer to assess the tradeoffs of using a collage metaphor in the design of the system. Along with other claims about organizational methods, the designer can choose the best option. Through this structure, the ideas portrayed within the claim can be passed along and reused by other designers, making claims reusable design knowledge components.

A developer can make use of this claim in various ways. First, the designer can gain insight into what the collage metaphor is. Since claims are grounded in theories and observational studies, the designer can gauge the effects of using the feature in their own design. It is conceivable that one could think of an idea similar to the collage metaphor (without knowing what a collage metaphor is), but it is hard to imagine that the effects of the design feature can be readily assessed. With the claim tradeoffs (upsides and downsides) taken into account, not only do designers gain inspiration, but also get the opportunity to consider whether integrating such a feature into their own design would be appropriate. During the evaluation phases of their work, claim tradeoffs are indications of aspects of their design that should be tested. Second, designers creating claims engage in the process of identifying specific design aspects that can be captured in terms of a claim. They are forced to consider theories and conduct their own studies to assess the effects of a feature, enabling designers to base decisions on sound reasoning instead of instinct.

2.3. Claims Library

In recent years, research within HCI has recognized that to facilitate reuse, claims must be generalized, classified, stored in a design knowledge repository, and retrieved when appropriate for use within a new design context (Sutcliffe, 2000)(Sutcliffe and Carroll, 1999). Within this literature, approaches for repositories of claims (referred to as claims catalogs or claims libraries) have been introduced, although core architecture and feature design are topics of much contention. We applied this vision of reuse in our own domain of interest: notification systems.

A growing class of applications is being developed to support multitasking information needs. Such applications, called notification systems, deliver valued information to users in a dual-task situation. Typically, a user conducts a primary task while they monitor information through a notification system as a secondary task. In this case, the goal of the notification system is to deliver valued information without introducing unwanted interruptions to the primary task (McCrickard et al., 2003). Instant messengers, large screen information exhibits, and car navigation systems are common examples of such systems.

Notification systems are characterized by three critical parameters: interruption, reaction, and comprehension (IRC) (McCrickard et al., 2003). *Interruptions* are events that reallocate attention from the primary task to a notification. *Reaction* is the rapid response to the stimuli presented by the notification system. Finally, *comprehension* is making sense of storing information in long term memory. Quantitative IRC values ranging from 0 to 1 can be assigned, establishing design goals and a method to empirically gauge how well the goals are met (Chewar et al., 2004)(McCrickard et al., 2003)(Chewar, McCrickard, & Sutcliffe, 2004).

When it comes to the design of notification systems, we see promise in being able to reuse notification systems knowledge. To support this cause, our previous work consisted of designing a digital library to store claims from the notification systems domain (Payne et al., 2003). Much work has been put into creating a formal structure for

claims to facilitate claims reuse. Currently, claims are stored along with several pieces of information among which are supporting scenarios, and associated IRC values (Payne et al., 2003).

The claims library is a collaborative environment in which designers can contribute claims from previous or ongoing designs or reuse claims for ongoing and future designs. The purpose of the claims library is to provide meaningful design knowledge and encourage designers to search for and consider claims about systems similar to an application they are currently designing. As designers retrieve claims, they can assess the design tradeoffs of possible artifacts. Supporting scenarios provide context to the use of claims and IRC values offer insight into the implications of the claims on notification systems.

The mere existence of a library, however, does not mean designers will reuse the knowledge contained within it due to factors such as content and searching. As a repository, it is good at storing the knowledge, but we believe it has greater potential in ensuring designers have the ability to reuse the knowledge. We strive to improve the model of the library to motivate HCI designers to reuse during their design process with the help of relationships.

2.4. Using Relationships

Looking beyond HCI and into the architecture domain we see the work of Alexander, the creator of a design knowledge form called design patterns, which are now prevalent within the software engineering community. Patterns describe common problems and establish possible ways to solving them. In his seminal book, (Alexander, Ishikawa, & Silverstein, 1977) patterns from the architecture domain range from higher-level city concerns, such as the distribution of towns, to lower-level issues, such as individual rooms in homes. Each pattern identifies problems regarding these issues and presents a solution. The solution itself can comprise of other patterns. Hence, patterns are linked together to point to other patterns. Following these links within the book can lead readers to other patterns that they may need, creating a browsing mechanism. Although Alexander does not use relationship types, his notion is similar to the core of this paper.

Zimmer (1995) builds on Alexander's work by attempting to apply relationship types to design patterns. He understood the notion that patterns were being linked to and even combined with each other without defining the nature of the association. He subsequently categorizes his relationships into three types: a pattern using another pattern, a pattern similar to another pattern, and a pattern combined with another pattern. While the relationships are useful, they are not expansive enough to cover the various activities of design. Most interesting is his understanding of the fact that the assignment of relationships is not an easy matter.

When it comes to digital libraries, those that exploit relationships are nothing new. Embly (1987) demonstrates a library containing abstract data types (ADT) for reuse that is structured to use automatic and user-defined relationships among entities. The objective is to locate ADTs in the library and use them in software that is under

development. The structure of the relationships facilitate locating using automated functionality, browsing by following links, and software building related activities through integration, providing increased flexibility. Unfortunately, the nature of the relationship types do not necessarily allow for a goal-oriented search strategy, although many of the concepts behind the design of the library are indeed useful.

Creech, Freeze, and Griss (1991) focus on how to structure and efficiently select components in a reuse library. They consequently explore the use of hypertext in selecting reuse components. Their belief is that the appropriate use of hypertext to structure components would aid the selection process. By providing a graphical view of a library, users were allowed to browse through the library by navigating from component to component. Although components were related to each other, the nonexistence of relationship types effectively meant there was only one possible relationship, limiting the degree of organizational and navigational improvements.

The most notable mention of relationships similar to our work can be found within the literature of claims. In the past, there have been assertions that for claims to be an effective component of reuse, they need to be classified and organized in a library. Previous work mentioned that there is a need for claim relationships within such a library to help bring structure and organization (Sutcliffe & Carroll, 1999). It is argued that relating claims can expose different levels of granularity and associating claims together can lead to the creation of new artifacts expressed as child claims (Sutcliffe, 2000).

Throughout these examples of uses of relationships we see how the need for linking pieces of design information together and understanding the nature of the link arises. They show how relationships are crucial to creating effective and usable libraries of reusable information. Through the application of relationships we can begin to see their benefits for developers and design.

3. IMPETUS FOR REUSE IN HCI

To instantiate a reuse solution within HCI it is essential to analyze the definitive solution from software engineering. Krueger (1992) successfully argues his own perspective on reuse. Many use this work as the de facto definition of reuse and its characteristics (e.g., (Ye & Fisher, 2002)(Basili, Briand, & Melo, 1996)(Johnson, 1997)(Sugumaran, Tanniru, & Storey, 2000)). Krueger suggests any approach to reuse must, "provide natural, succinct, high-level abstractions that describe artifacts in terms of 'what' they do rather than 'how' they do it (Krueger, 1992)." It is more important to first describe components in terms of what they mean to an overall design instead of what they contribute. For example, it is more beneficial if a designer knows a certain component is a potential solution to a design problem rather than knowing what the solution described by the component is. Our conceptualization of *design knowledge relationships*, the characteristics describing interactions between knowledge components, forms the basis of our approach to describe 'what' components do—a topic covered in Section 4.

Krueger's broad analysis of reuse techniques demonstrates how each technique is based on four aspects crucial to facilitating reuse: abstraction, selection, specification, and integration. *Abstraction* is an essential feature to any reuse technique. The essence of abstraction is a succinct description concealing the unimportant details and only showing the most important ones. Without it, designers would be forced to meticulously peruse through assemblies of reusable components, prolonging the time spent looking for an appropriate component. *Selection* involves locating, selecting, and viewing an artifact for potential reuse. Classification or categorization methods may be used to organize a reuse library for artifacts to be found and reused by designers. In many cases, general components may need to be adapted or further specified, accentuating the need for *specification.* This is often an important step because designers may not always find exactly what they need and will be forced to adapt components. Finally, components must be gathered and *integrated* into a coherent design. Designers must understand how components that have not been used together before can interact with each other within the context of the overall design. Section 5 introduces how our concept of design knowledge relationships can be applied in terms of these four reuse aspects.

We establish the need for the reuse aspects by asserting they map to problems associated with our own claims library. Understanding these problems and associating them to all of the reuse aspects allows us to demonstrate a software engineering approach has the potential to help HCI knowledge reuse. To the best of our knowledge, we have not seen any other example of Krueger's vision being brought into HCI research, giving us the opportunity to demonstrate what we can contribute. We continue this section by presenting four problems and demonstrate how they can be solved with the incorporation of the reuse aspects.

3.1. Problem 1: Abstraction for Identification of Component Types

While the structure of a claim is very simple, claims can be written and used in a variety of different ways, leading to claims of different types. This raises the need for the ability to identify such types. However, the problem is not simply solved by labeling each claim. The reason why context is so important is because the type of a claim depends on the context of its use.

In SBD, a prominent distinction is made between the problem and design domains. In the problem domain, designers explore the reasons for design through requirements analysis. In the design domain, designers begin to create actual design features to satisfy requirements. Claims can be used in both the problem and design domains. Problem claims depict artifacts that are deficient within the current method of accomplishing a task. Design claims are created as solutions to problem claims, portraying the future design. A claim could be a problem claim in one design, but a design claim in another.

Norman (1986) presents a cognitive engineering model in which users cycle through two key obstacles during interaction: the Gulf of Execution and Gulf of Evaluation. In the Gulf of Execution, a user interacts with the system to carry out a determined action. In the Gulf of Evaluation, users assess the current state of the system. Similarly, claims can be written to support either of these two Gulfs.

The problem and design domains and the two Gulfs create the basis for interactive system design in SBD. The type of a claim is determined by which two of these four domains a claim fits into. Its type, however, can change depending on the context. For example, a certain claim may be a problem claim in the Gulf of Execution, but in a different usage context be a design claim in the Gulf of Execution. The key problem is that a claim fits into two of four domain types. Such cases illustrate the importance of context, making it imperative to be able to distinguish between these types of claims. Designers using the library must be able to quickly identify which two domains a claim fits into without having to analyze the details of the whole claim and make a judgment. Our solution is to incorporate the use of abstraction. Section 5.1 outlines how we succinctly describe claims to identify their types.

3.2. Problem 2: Selection of Library Contents

When we think of a traditional search, we imagine ourselves providing a query. The query is typically a keyword and/or a value for an attribute. Once entered, search results are displayed. If the user does not find what is needed, he/she is forced to go back and reformulate the query. There is an inherent problem in this process. The user is forced to anticipate the contents of what they are searching within. Without knowing what is available, users go through the trial and error procedure until they find what they need.

The solution to this problem is to make the contents of a library visible to the user. Users of a repository must not be forced to guess what is available, but should rather be able to browse through the contents in order to develop an understanding of what the repository can provide them with (Godin, Pichet, & Gecsei, 1989).

Simply allowing users to browse a library is not enough. As designers browse, they may often find that an artifact does not quite fit their needs. Being able to search for an alternate artifact that is close to the general idea of the artifact already found is an important task that must be supported. To find the most appropriate claim, looking at claims that are within the same design context will give the designers a higher chance of finding what they need. Therefore, searches should build upon previous searches to develop the context the designer needs. Maintaining an established context during the search for a claim should yield better results in terms of finding the claims the designer need. To make the contents of the library visible to the user browsing capabilities must be increased. Through the process of selection, designers should be able to discover new claims in the library and lead to further alternatives that may be of use. In Section 5.2 we describe how selection allows designers to gain an understanding of a claim and other claims that are within the "vicinity" through networks of claims.

3.3. Problem 3: Specification to Adapt Components

We can all accept the fact that a reuse library cannot contain all the components that a designer will need. Even after searching for alternative components, the designer may still not find exactly what is needed. It is very likely the designer will try to get as close as possible to what they need and then stop searching.

Because of this, there must be a way to adapt components so that they become a perfect fit for the intended design. This will often lead to the creation of a modified component. The same is true for claims. Designers will want to modify claims they find in the claims library to suit their own needs during their design process. The challenge is to create the claim correctly and include it within the library so it too can be reused by others. This is key in creating a cycle of reuse, where new components are created through reuse and then reused by others. Through specification designers can adapt components according to their needs for their own designs. Section 5.3 outlines how designers can specify new claims and include them in the claims library.

3.4. Problem 4: Integration of Reused Components

It is often difficult to find different artifacts and be able to combine them into a coherent design. When designers are done gathering claims from the claims library, they must begin to think about how all of the claims will interact with each other. It is quite likely that most of the claims they gather will be claims that have previously never been used together.

Without understanding how claims will interact with each other, a designer is forced to consider a series of ideas without getting a view of the overall concepts behind the design. There is a need for a design work product developers can constantly refer to once major conceptual design efforts have been completed. Additionally, it should represent the current state of the design and provide insight into future design work. The process of collecting and combining the claims is comparable to the process of integration. We propose the idea of a claims map in Section 5.4 to address the issues of integrating claims.

4. CLAIM RELATIONSHIPS

We have established the link between problems in the claims library and Krueger's reuse aspects in Section 3. Building on the message of providing, "natural, succinct, high-level abstractions that describe artifacts in terms of 'what' they do rather than 'how' they do it (Krueger, 1992)," we propose the use of relationships as an instrument that will serve this purpose. Relationships between reusable components can provide the succinct descriptions advocated by Krueger.

To implement solutions to the outlined problems, we propose the use of *claim relationships* (Wahid et al., 2004a). These relationships show how pieces of HCI design knowledge can be linked together to describe the nature of their connection. Thus, designers can view claims in terms of problems, solutions, alternatives, sequences, and combinations instead of their internal details. Each relationship is used to succinctly describe the nature of 'what' a claim does for an overall design. One can explain the larger implication of a claim for a design, leaving until later the specifics of what is described within the claim itself.

Claims are well-situated within HCI because they describe the psychological effects of design features. Claim relationships must respond to the need for richer descriptions

of their use within design processes (Rosson & Carroll, 2002)(Norman, 1986) and their structure. Two of the relationships we will define are closely tied to the most fundamental process-related design steps followed in Scenario-Based Design. The remaining relationships describe how a claim as a whole or its upsides and downsides can interact with other claims. These two forms of descriptions make the relationships unique to HCI, portraying a plethora of methods to amalgamate HCI knowledge.

4.1. Postulating/Predicating Claims

The first key relationship type between claims is the postulation/predication relationship apparent in the process of mediated evaluation. In a claims analysis, a designer assigns credit or blame attributions to artifacts, which are continuously refined in subsequent design activities. Design activities in SBD typically iterate through three processes, from requirements analysis to general activity design to specific design of features—a pattern paralleled by the themes addressed in each claims analysis. In each process, a designer collects evidence to assert postulating claims to guide the next process, while alleviating or refuting claims from the previous process with predicating claims based on new ideas or evidence. Each process acts as a problem that the next process must solve. Thus, there is a natural relationship between problem and design solution.

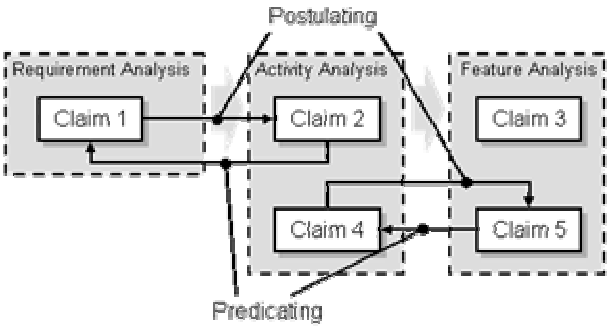

Figure 2. Postulating/Predicating Claims

As illustrated in Figure 2, a designer would create Claim 1 to express the aspects of the problem domain based on requirements analysis. This claim can then lead to two new claims in the activity analysis phase (Claim 2 and 4) through postulation to solve the problem depicted by Claim 1. The relationship in the opposite direction is a predication. The two claims in the second phase lead to Claim 3 and 5 in the feature analysis phase through further postulation.

Maintaining awareness of open windows through a taskbar
+ Allows users to recognize that a window is currently open
+ Enables users to get information at their own will
- Does not provide a count of the number of windows that are open
- Requires that the user know that taskbar items are associated to windows

Relaying information through changing text

+ Allows more information to be displayed in limited space

+ Constantly updates user on the current status of information

- The amount of text visible at any point in time may be too little for a user to understand the message

- The rate of the change in text may be too fast, not allowing users to read or see

Figure 3. Two claims that have a postulation and predication relationship between them.

To better illustrate the postulation and predication relationships, consider the two claims in Figure 3. The first claim depicts a problem situation describing the current method of maintaining awareness. We use the second claim to solve the problem in the new design. Thus, the first claim has a postulation relationship with the second. A predication relationship exists in the opposite direction. This association is critical to exhibiting the problems and design solutions that are identified by designers during the design process.

4.2. Executing/Evaluating Claims

As previously mentioned, Norman (1986) presents an argument for interface design as a cognitive engineering discipline, where designers assist the user with progressing through stages of action. Each stage falls within one of two Gulfs—the Gulf of Execution (where the user executes after deciding upon goals and specific action sequences) and the Gulf of Evaluation (where the user appraises the current state of a system). The SBD methodology describes how information design decisions influence the stages of action required for crossing the Gulf of Evaluation, and how interaction design addresses the Gulf of Execution (Rosson and Carroll, 2002). In information design, interface choices such as use of color, animation, visualization techniques, and layout are made about specific features. Interaction design is more concerned with selection of controls, widgets, affordances, and input techniques.

Claims can be created specifically to fit a certain stage of action and, consequently, a certain Gulf. Certainly, a given artifact may be the subject of both Gulf of Evaluation and Gulf of Execution claims. It is helpful to have a relationship to describe this linkage. Some artifacts may only support the user in one of the Gulfs, but may typically be used with other artifacts that address either the same or opposite Gulf. Therefore, the relationship between two feature claims can be described according to the "destination claim." A destination claim in the Gulf of Execution can be the executing claim for claims in either Gulf. Likewise, a claim in the Gulf of Evaluation could be the evaluating claim for others in the same or opposite Gulfs. Figure 4 demonstrates the use of this relationship with respect to the Gulfs.

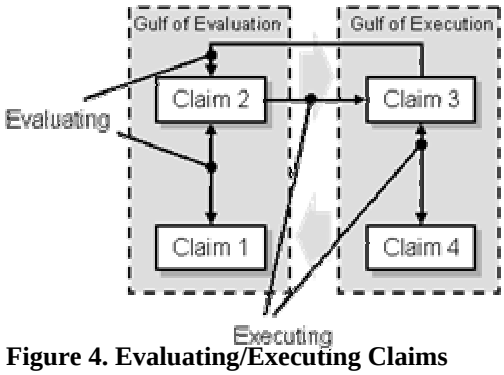

Figure 4. Evaluating/Executing Claims

The order in which the claims are placed into the stages of action determines the user task flow. As a user progresses through the stages of action, one can identify which claim comes into play. In Figure 4, the user's task flow follows the numbering of the claims. If desired, a task flow can be explicitly represented using the relationships in just one direction, eliminating the use of relationships in the reverse direction.

Recognition of the taskbar for information delivery
+ Spacing between taskbar items allows identification of a single piece of information
+ Flashing taskbar items allow users to quickly recognize the taskbar
- Flashing taskbar items can cause unwanted interruptions

Clicking on a taskbar item to switch active windows
+ Aids in the switching primary tasks
+ Can increase comprehension
- Causes a greater primary task interruption due to reallocation of attention to new primary task
- Previous active window may completely disappear

Figure 5. A Gulf of Evaluation claim and a Gulf of Execution claim.

In Figure 5 there are two claims that fit into the two Gulfs. The first claim belongs to the Gulf of Evaluation since it describes a situation in which the user monitors the state of the interface to receive the updates. The second claim is a direct interaction method and consequently is a Gulf of Execution claim. When moving from the first claim to the second, we have an execution relationship. The evaluation relationship exists in the opposite direction.

4.3. Generalizing/Specifying Claims

Claims can have different scopes depending on the granularity of the artifact components which they describe. A general claim might describe psychological effects that result from the holistic design or several distinct portions (combinations of widgets) used in a variety of contexts. General psychological effects can be elaborated by claims that have a narrower scope. These claims apply to very specific parts of an interface (a

particular button), usage instances, or user characteristics. They are most useful in guiding component reuse, since they describe an interface at its finest detail and raise in-depth issues related to the interface. However, the "general idea" of a specific claim will often have more frequent applicability to new design problems.

Sutcliffe and Carroll (1999) propose a factoring method for evolving between claims of different scope and use the terms "parent claim" and "child claim." In our framework of claim relationships, the generalization/specification relationship is the linkage between two claims with different scopes (see Figure 6). A generalizing claim is the consequence of taking a specific claim and generalizing it to apply to a courser artifact or usage context granularity. A specifying claim is the opposite, in that it is the result of narrowing the scope of a general concept. The process of generalizing allows one to create claims applicable to many situations (see Figure 7). This course of action permits one to take ideas from a specific problem and reuse them in a new context to solve design issues—sowing the seeds for innovation and technology transfer. A key concern in generalizing and specifying new claims is with extending or narrowing the scope in an invalid manner, thus, losing the support of empirical or theoretical evidence grounding the original claim. For example, a generalizing claim can only be reliably used in a narrower context, as it inherits upsides and downsides characteristic to specific conditions.

Relaying information through changing text

+ Allows more information to be displayed in limited space

+ Constantly updates user on the current status of information

- The amount of text visible at any point in time may be too little for a user to understand the message

- The rate of the change in text may be too fast, not allowing users to read or see

Cycling banner for relaying text in desktop secondary displays

+ Allows more information to be displayed through cycling in a limited amount of space on the desktop

- Transitions between information may be too distracting due to secondary displays being in view most of the time

Figure 6. An example of a general and specific claim.

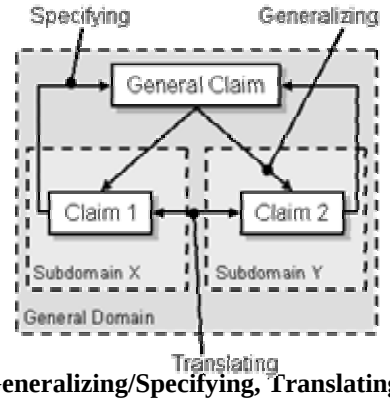

Figure 7. Generalizing/Specifying, Translating Claims

4.4. Translating Claims

Existing claims may not be directly applicable to new design problems. They may describe problems that are similar, but not completely accurate. Often though, existing claims provide the basis for the generation of new claims due to recognized similarities between the current problem domain and the one in which the original claim exists. The relationship from the original claim to the new claim is called translation (see Figure 7). Translating claims are typically alternatives to each other with the same scope. The relationship can also indicate where cross-domain reuse has occurred in the development of a system.

Use of fading between information transitions
+ Decreases interruptions caused by dynamic information changing
- Transitions between information may be too distracting if transitions are too fast

Using scrolling to display new information
+ Allows the same amount of space to be used to show more information
- The speed of the scrolling may be too fast, hindering reading
- Transitions between information may be too distracting

Figure 8. An example of two claims with a translation relationship.

The crux of translating is the establishment of a correlation between the existing claim and the claim to be created. To accomplish this, the designer is required to consider the existing claim at a deeper level of abstraction, or a generalized version of the claim. While no explicit generalized claim is created, as suggested by Sutcliffe (2000), the general form of the original claim exists in the mind of the designer. Then, the specific aspects of the original claim are altered to fit its new context of use, thus creating a new translating claim. Ideally, many of the original tradeoffs will still apply in this new context, however, situating the claim requires reevaluation of upsides and downsides with respect to this context.

4.5. Fusing/Diffusing Claims

The fusion relationship between claims is the outcome of the combination of two or more claims into a new fusing claim. A developer recognizes certain aspects of various claims can be applied together in a new and innovative way, such as Claim 3 in Figure 10. The result is a hybrid claim that is pieced together with artifacts and design rationale from each of the supplemental claims. In addition, further design rationale may be required due to novel application of the original artifacts.

Similarly, a designer could break a claim into smaller claims, taking only a fraction of what exists in the original claim to produce a diffusing claim (e.g., Claims 2.1 and 2.2 in Figure 10). This time, the designer focuses on part of a larger claim and elaborates on artifacts and tradeoffs that pertain to the new, smaller claim. This practice may result in the creation of multiple smaller claims, depending on how the original claim is divided (i.e. there were equal acting parts of the original claim). This relationship between the original super-claim and the resulting fractional claim is called diffusion. Figure 9 shows an example of two claims that are fused together to create a new claim.

Mouse-over changes color of clickable text
+ Allows user to understand they may click on the text
- May imply that less information is shown, leaving most of the information to be shown after a click

Red highlighting for urgent or important information
+ Allows users to quickly recognize urgent information, drawing the user's focus
- Requires a systems ability to determine what information is considered urgent

Mouse-over changes color of clickable text to red showing inaccessible information
+ Allows users to understand whether they should click on the text
- Forces user to mouse-over the text in order to gain the information regarding the status

Figure 9. The first two claims are fused together to create the third claim. The third claim can be diffused into the first two claims.

Relating claims in this manner can illustrate progress throughout design iterations as well as where claim reuse has occurred. During the design process, testing and evaluation provide the basis for the validation of claims. Another result of this process may be the fusion of two claims that seem to demonstrate strong positive results in combination or the diffusion of a claim that exhibits distinctively different results for different aspects of its makeup. Additionally, two existing claims from completely different problem domains may be fused into a new and innovative claim. This process was noted, but not named by Carroll and Kellogg (1989). An intermediate step, similar to the generalization process described above, requires the designer to consider "what" the claim does, as

opposed to "how" this is accomplished. This distinction depends on the level of abstraction at which the claim is considered.

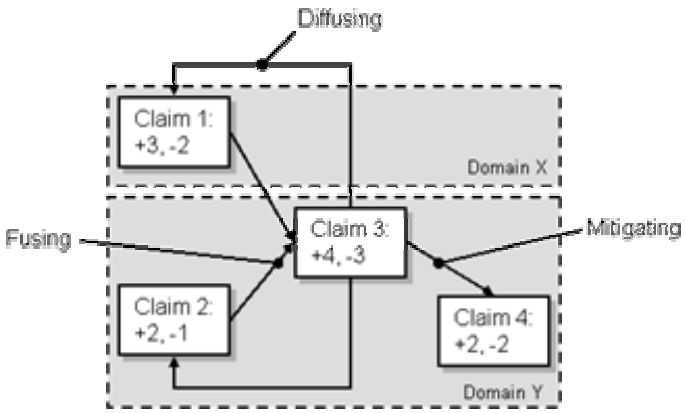

Figure 10. Fusing/Diffusing, Mitigating Claims

4.6. Mitigating Claims

The strength of a claim relies on the explicitness and poignancy of its upsides and downsides. Upsides can represent the potency of an interface, while downsides dictate adverse consequences resulting from the interface design. Explicitly identifying weaknesses of a design often expedites improvement of usability—a process that should be repeated as new flaws are uncovered.

Scenarios are descriptions of a sequence of mental and physical actions a user of an interface may go through. Carroll (1994) suggests that one can use scenarios to construct new alternative scenarios. The process of analyzing the psychological design rationale within a scenario allows designers to identify alternative scenarios which may be appropriate for other possible usage scenarios. Alternate scenarios are created in a way such that they can handle or correct disadvantages and at the same time maintain or improve strengths of other scenarios. Through multiple iterations, scenarios can be developed and perfected over time.

This same process is valid for claims. A mitigation relationship is the result of a process in which a new claim is created to manage limitations of another claim (see Figure 10). As previously mentioned, claims make their downsides explicit, clearly identifying areas for which designers must also find solutions to problems they believe are of a considerable magnitude. The purpose of a mitigating claim is to resolve another claim's downside to improve the overall design. The method of creating mitigating claims can be repeated as many times as needed until designers are satisfied. After designers make improvements to an interface in a design iteration, usability testing must validate the improvements by testing the performance of the mitigating claims. Thus, mitigating claims become a trace of the design improvements that are made over time. Figure 11 shows an example of a claim whose second downside is mitigated by the second claim.

Cycling banner for relaying text in desktop secondary displays
+ Allows more information to be displayed through cycling in a limited amount of space on the desktop
- Transitions between information may be too distracting due to secondary displays being in view most of the time

Use of fading between information transitions
+ Decreases interruptions caused by dynamic information changing
- Transitions between information may be too distracting if transitions are too fast

Figure 11. The downside of the first claim in mitigated by the second claim.

5. REUSE ASPECTS AND CLAIM RELATIONSHIPS

How do we use the claim relationships to enact Krueger's cause? Claim relationships can be used in different ways to manifest solutions to our problems. Depending on the context of their use they can exhibit characteristics of each reuse aspect. The use of certain relationships can help abstract claims to identify their type. The relationships can be used to form a network of claims to aid selection. When needed, a designer can create a new claim based on a single relationship. Finally, claims can be integrated together using the relationships. In this section we expand on these applications and present studies conducted on their use.

5.1. Abstraction

Abstraction is the first aspect of reuse to consider. Although it is a very simple notion, one needs to incorporate its use to avoid forcing designers from continuously having to read the details of a complete claim (Krueger 1992). Designers must be able to identify the correct type of claims crucial to the high level design of their system. Through this identification process, they can begin to establish a sense of context within their search process.

Laurian is a designer who wishes to create a new desktop notification system that will allow users to monitor the news headlines throughout the day. Employing a Norman-style design methodology, she wants to first identify problem claims—situated within stages of the Gulf of Evaluation and Execution—then consider design claims that address the key issues of each problem claim. Knowing the claims library contains many design claims, she decides to search for claims to reuse using a standard keyword text search. She provides queries that are relevant to her design domain and analyzes each result, but many designs within her domain of interest have little to do with the problems she identified. She spends a lot of time reading irrelevant claims and gets frustrated when she can not find relevant claims for her issues at hand.

Figure 12. A scenario showing the need for abstraction in reuse.

To tackle this problem of reuse, we created claims in the claims library and assigned relationships. We allowed users to see the title of related claims and the relationship type while viewing the details of claims. This strategy created a network of claims designers can traverse through by following relationships, a key element in demonstrating the use of abstraction.

In Section 3.1 we described four main domains that form the basis of a design created through SBD: problem, design, execution, and evaluation. During the design process, a designer using claims to create a system should first identify all the problem claims in the Gulf of Evaluation and the Gulf of Execution through the process of requirements analysis. They should then proceed to find all the design claims for both the Gulfs based on the identified problems. This overall procedure requires that they find the appropriate type of claims for each domain when using the library.

We adapt these four domains and use them as the basis for abstraction by allowing users of the library to identify an appropriate claim type. In abstraction, it is important to see only the most important data and hide the rest of the details (Krueger, 1992). In this case, when looking at a claim, the only data a user sees for the related claim is the claim title and the relationship type.

The postulation/predication and the execution/evaluation relationships are the most important relationships when it comes to supporting abstraction in the claims library. For this reason, they are referred to as the high level relationships. These relationship types allow the user to identify which domains a claim would fit in. For example, if a user looks at a claim's postulating relationships, they will be able to understand that within the current context the claim they are looking at is a problem claim and that the related postulating claims are design claims. This same idea applies to the execution/evaluation relationships. Designers will be able to understand what Gulf the claim should be placed in. Through this simple process, a designer can quickly begin to recognize and locate the correct type of claim that is needed as they simultaneously progress through their design

processes (Wahid et al, 2004b). A user study investigating this aspect (described in further detail in Section 5.2) proved designers can successfully identify claim types within an established current context.

Once a claim is abstracted, the designer must focus on finding the most appropriate claim for their system by browsing. Our next section on selection describes how the claim relationships can be used as a browsing mechanism. We present a study on the use of selection and expand upon the role of abstraction.

5.2. Selection

Unfortunately, as with most knowledge management systems, acquisition is the bottleneck (Wagner, 1990). Component selection is a very important characteristic of reuse (Krueger, 1992) (Dusink & van Katwijk, 1995), however, the current state of the retrieval mechanisms in such reuse repositories is inadequate. Searches for components are often limited to keywords and classification schemes, such as tasks and IRC values in our case, which only serve as parameters to enhance retrieval. Browsing capabilities to navigate from one component to another are nonexistent or also inadequate in such systems. Furthermore, most design knowledge repositories do not support an outlined search strategy, a series of steps one can follow depending on their needs to ensure that they will find all of the components they need.

> Laurian decides to use the claims library to search for claims that will provide her with design ideas for her notification system. She uses the keyword text search to look for claims that may help with a specific part of her system. Unfortunately, the search does not return any results due to her query being too specific. She reformulates the query over several iterations and begins to get some results, but unfortunately the results are not relevant to her design. She once again begins to get frustrated because she can not find any relevant ideas to inspire her design. She has no sense of the contents of the claims library, and the library lacks mechanisms to familiarize designers with its contents.

Figure 13. A scenario demonstrating the need for improved selection in reuse.

Generally, users either search or browse digital libraries (Blanford, Stelmaszewska, & Bryan-Kinns, 2001). A prominent application of the relationships is to use them to aid selection by creating a browsing mechanism (Wahid et al, 2004b). While this solves our second problem of showing what is available in the library, it also establishes a search strategy for finding appropriate claims (further solving our first problem), and alleviates the acquisition bottleneck.

Krueger (1992) describes the selection aspect of reuse in terms of three key concerns of selection: classification, retrieval, and exposition. When selecting, a user must first be able to classify the components. This is similar to understanding the type of the

component. Once classified, the user must retrieve the component and then expose the details of it. We proceed to describe how claims relationships can be used to browse our library and present the results of a study evaluating the use of selection.

CERVi

Before claim relationships were fully integrated into the claims library, to explore the selection aspect of reuse, the Claims Exploration of Relationships Visualization (CERVi) tool was created to improve knowledge acquisition through the use of claim relationships. Similar to how the relationships are currently implemented in the claims library, the tool allows designers to find claims by navigating through a network of claims connected by claim relationships. In the tool's case, users are using a visualization of the claims library. Users can find appropriate claims by analyzing related claims, providing recommendations based on the context of the visualized claim.

The tool is designed to enhance the browsing experience and to move away from the traditional idea of searching for claims using a query. With the introduction of the tool, our studies showed that evaluators are able to find appropriate claims and create more accurate conceptual designs by using the relationships as a search strategy while browsing

CERVi Search Strategy

During the design process, designers following the SBD approach typically locate relevant problem and design claims for each of Norman's stages of action to create a well-planned design. CERVi allows designers to identify problem and design claims and place them correctly within the Gulf of Execution or the Gulf of Evaluation.

Since the process of reuse starts with abstraction, the postulation/predication and execution/evaluation relationships (high level relationships) naturally form the basis of a search strategy. They enable designers to navigate between problem and design domains and between the two Gulfs. The remaining relationships are called the low level relationships. Instead of allowing for navigation, these relationships allow the designer to focus within a certain area of the design to find more relevant claims. The overall strategy for searching supported by CERVi is first to use the high level relationships to find claims that may fit into the overall design and then to use the low level relationships to find the most appropriate claim (see Figure 14).

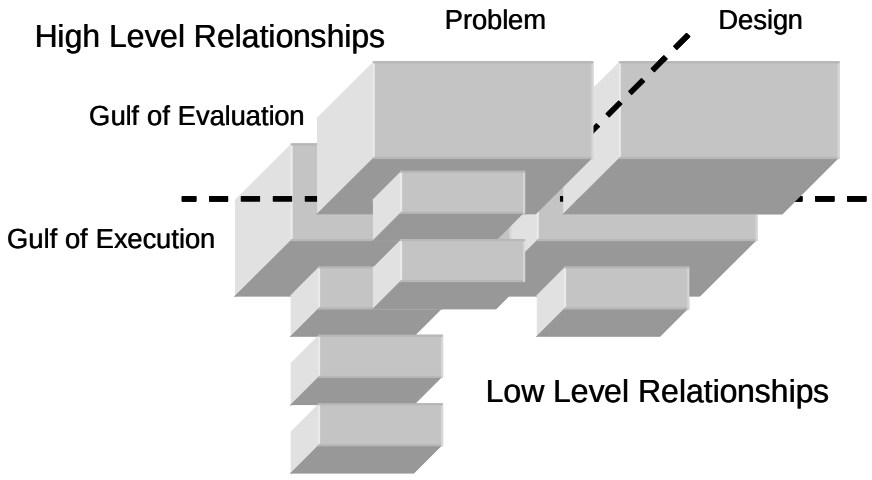

Figure 14. The high level relationships allow movement between domains. The low level relationships allow designers to focus in a particular domain.

Evaluation

Emerging from the project highlights, our user testing goals were to determine the impact of claim relationships on finding reusable knowledge and to validate the process as a selection mechanism. We decided the best way to test this was to ask designers to create designs by collecting claims. To accomplish these goals, we developed a design table with the intention that it would serve as a tool to outline a conceptual design in terms of claims. The table consisted of two rows, one corresponding to the problem domain and the other to the design domain, and six columns, each representing one of Norman's stages of action.

We asked 15 undergraduate HCI students to read a given scenario outlining the need for a notification system. Their task was to describe a notification system that would satisfy the scenario by collecting claims and placing them in the design table. They were allowed to search for claims using the claim relationships through CERVi or use a traditional search with queries in any combination to locate at least ten claims that were relevant to the design they envisioned. Descriptions of each stage of action and definitions of each claim relationship were given to allow them to understand how to leverage their utility. Several claim ID numbers from the claims library were included under various stages in the design table to initiate a search for claims, while other portions of the table were left blank. Upon completion of the task, participants were asked to describe their envisioned design and complete a survey. Questions were designed to determine the incorporation of those relationships into participants' search strategies and the impact of the relationships on the resulting design work.

We decided to conduct a second round of testing using 6 graduate HCI students who were equally unfamiliar with the concept of claim relationships. They were given the same design table and scenario and asked to design a notification system. The only difference in the second round was the inclusion of a ten minute presentation defining

each claim relationship and showing how to use the relationships to find claims for specific parts of the design table.

We hypothesized our selection mechanism would result in student designers collecting appropriate claims and placing them in the correct context, thereby resulting in better conceptual designs based on claims. We define a claim's state of being appropriate as being well suited for the given scenario and the student's stated design vision. Correct context for a claim is established when it is placed in the correct domain (problem vs. design) and stage of action in the design table.

Results and Discussion

To analyze the results, we calculated the number of errors that were made in the design table based on a predetermined solution set (see Table 1). For each design table we recorded the number of correctly placed claims, claims in the wrong Gulf, claims in the wrong stage in the correct Gulf, and claims in the wrong domain. We then calculated a design score and percentage for each participant. While the weights of incorrect Gulf and domains were 1.0, errors regarding the stages were given a 0.75 weight because claim relationships do not distinguish between stages of action.

Most of the students in the first round of testing used CERVi more than the regular search method and consequently retrieved more claims using the tool. Among the fifteen students, however, only two designs received a score of more than 80%. Most of the errors were committed due to placing claims in the wrong stage. The mean design score percentage was 39.30% with a standard deviation of 0.29. In general, this group did a poor job of establishing the correct context for the claims they put in the design table. Although the majority of the errors came from placing the claims in the correct Gulf, but incorrect stage of action, we were surprised to see an almost equally high number of Gulf errors. Errors in domain placement, however, were lower. In light of these results, several students were called back for an interview. Some students thought they understood many of the concepts behind claim relationships. They realized they missed many concepts when their errors were pointed out, but were surprised at how easily they could understand everything when explained.

The graduate students did extremely better than the undergraduates. The initial presentation encouraged all the participants to use CERVi. There were four designs that scored over 80% with an overall average design score of 72.86% and a standard deviation of 0.23. A majority of their claims were placed correctly in the table and errors were notably lower although the highest error rate was still in incorrect stage placement.

We attribute the observed difference in design scores between the two groups to the existence of a small learning curve. A 10 minute presentation was all that was needed to increase the design scores. We believe this is a small price to pay for the benefits that have been observed in the second round of testing. Gulf and domain errors are indicators of how well participants can abstract using the high level relationships (see Section 5.1). The notable decrease in these two error types in the second round demonstrate that users can indeed successfully abstract using the high level relationships. Errors regarding the

stages do occur because claim relationships do not distinguish between stages, but with the ability to identify the correct Gulf, we believe the chances of identifying the correct stage are increased. These results demonstrate designers can indeed place claims in the correct context.

Participant	Correctly Placed Claims	Claims in Incorrect Gulf	Claims in Incorrect Stage	Claims in Incorrect Domain	Design Score	Design Score Percentage
First Round: Undergraduates						
1	6	0	2	2	2.5	60.00%
2	0	6	2	1	-8.5	0.00%
3	3	3	3	1	-3.25	30.00%
4	1	3	4	2	-7	10.00%
5	2	3	4	1	-5	20.00%
6	2	2	4	3	6	18.18%
7	3	5	0	0	2	37.5%
8	**8**	**1**	**2**	**1**	**4.5**	**66.66%**
9	**8**	**0**	**0**	**0**	**8**	**100.00%**
10	**6**	**0**	**3**	**0**	**3.75**	**66.66%**
11	**3**	**0**	**7**	**1**	**-3.25**	**27.27%**
12	2	0	6	1	-3.5	22.22%
13	1	2	0	2	-3	20.00%
14	**8**	**0**	**1**	**0**	**7.25**	**88.88%**
15	2	6	0	1	-5	22.22
Mean	3.66	2.06	2.53	1.06	-1.36	39.30%
St. Dev.	2.76	2.21	2.19	0.88	5.22	0.29
Second Round: Graduates						
16	**13**	**0**	**1**	**0**	**12.25**	**92.85%**
17	**9**	**0**	**2**	**0**	**7.5**	**81.81%**
18	3	3	3	1	-3.25	30.00%
19	**8**	**0**	**2**	**0**	**6.5**	**80.00%**
20	7	2	2	0	3.5	63.63%
21	**8**	**0**	**1**	**0**	**7.25**	**88.88%**
Mean	8	0.83	1.83	0.16	5.62	72.86%
St. Dev.	3.22	1.32	0.75	0.40	5.17	0.23

Table 1. Results from the design tables showing the number of correct claims, errors, and corresponding design scores for both rounds. Designs chosen to be the most appropriate have bold participant numbers.

The second form of analysis done was a qualitative assessment of how appropriate the chosen claims were for the design. The appropriateness of a claim was judged based on how well the overall concepts within a claim relate to the design. Our predetermined solution set contained all the claims we previously judged to be appropriate for the given scenario. As domain experts, we still analyzed the participants' envisioned design descriptions and their chosen claims. Claims were judged to be appropriate if they

appeared in our solution set and were related to the declared design vision. In the event a claim did not appear in our solution set, we relied on the design vision to judge whether the participant did choose a claim that would suit their design.

We judged 5 designs from the first round and 4 designs from the second round to have claims sufficiently related to the scenario and stated design visions. All of these designs were created with the use of CERVi, showing the use of claim relationships can result in the retrieval of more appropriate claims given the claims are available. Our most interesting choice was the design created by participant 11. Although the design score was extremely low due to many stage errors, the claims were good choices and depicted important aspects of the design.

Based on Krueger's (1992) definition of selection and its application to reuse, we asked questions that were designed to validate CERVi as a selection mechanism (see Table 2. Answers were recorded using a 5-point Likert scale, ranging from Strongly Disagree to Strongly Agree, and optional written responses. The questions addressed general selection issues as well as classification, retrieval, and exposition concerns within selection.

Question	Concern	First Round	Second Round
I was able to locate useful claims using the relationships.	General Selection	73.33% Agree	86.66% S. Agree
I was able to understand why two claims had the relationship they had.	General Selection	70.00% Agree	73.33% Agree
I used the given claim IDs as an index for locating more claims.	General Selection	93.33% S. Agree	90.00% S. Agree
The relationships gave me an understanding of what the related claim does.	Exposition	68.00% Agree	83.33% S. Agree
The claim's details were key in the selection of the claim.	Exposition	77.33% Agree	76.66% Agree
The relationships allowed me to understand where a claim could be placed on the table.	Classification	73.33% Agree	80.00% S. Agree
I was able to easily retrieve a claim based on a displayed relationship.	Retrieval	81.33% S. Agree	80.00% S. Agree
I understood the distinction between higher level and lower level relationships.	Classification	50.00% Neutral	73.33% Agree

Table 2. The questions asked related to specific characteristics of selection and the results of both rounds.

Responses to the classification questions, related to the conceptual structure of the relationships. Initially, responses indicated a weak understanding of the distinction between high and low level relationships and uncertainty in using those relationships to place claims correctly within the design table. This sharply changed with the second group which pointed out they had a good understanding once they received an explanation about the selection mechanism. Responses to the retrieval questions, related

to the design of CERVi, showed that participants could easily retrieve a claim based on a displayed relationship. Answers to questions related to exposition indicated examining claim details was a key aspect of the selection process.

The results of our evaluation show our approach of using relationships shows promise in helping designers and, consequently, designs. With this work we are able to say the application of claim relationships for selection is indeed a valid and useful application for finding claims. Through the use of the relationships, users gain an understanding of what is in the library without having to anticipate the content by creating search queries. Furthermore, the identification of the correct context allows designers to find appropriate claims, a need specified by the first problem.

The next section considers the case where a relevant claim is not found, accentuating the need to create a new claim. A practitioner can apply a claim relationship to specify a new claim. Two studies related to how the claim is created and what relationship should be used are presented.

5.3. Specification

Certainly our claims library cannot provide all the claims that are needed for any notification system design. Its contents are finite. It is quite possible users of the claims library may find a claim that is close to their needs, but not exactly appropriate for their design, forcing the designers to create their own claims when they see fit. Since this is inevitable, the question of how one can then facilitate future reuse of the new claim arises. Important in this investigation is understanding how claims are created and what relationship comes into play.

> Laurian searched the claims library and found a number of claims that are relatively appropriate for her design. Some of the claims she found are similar to what she needs, but are not completely relevant. Nevertheless, she is inspired by the claims and wants to create new design features for her own design. Based on several identified claims, she pieces together aspects of them to create a new claim that describes the tradeoffs of the design features she intends to use. While Laurian's own personal success is important, far greater benefit could be realized by adding the new claim—with any support for it that emerged through product design—into a claims library for others to find and reuse.

Figure 15. A scenario showing the need to specify components for use.

The act of adapting a reusable component to fit one's needs falls under the reuse aspect of specification (Krueger 1992). Hence, our third application for claim relationships is to use them as a specification method. Instead of creating a completely new claim, we advocate that the claim should be created by reusing a claim that is

reasonably close to the needs of the design. As the new claim is being created, a relationship should be chosen to guide the creation.

For example, if a designer finds a claim about the general use of color in notifications, they may choose to use it in their design. Of course such a general claim may be somewhat appropriate, but not the best fit. In this case, the designer may wish to use a more specific claim about color—a claim about using the color red to alert users for instance. When creating this new claim, the designer would in effect be creating a specifying relationship from the original claim to the new claim. If this relationship is explicitly noted when creating the new claim, the designer can easily modify the style of the claim, add the new claim to the claims library, and assign the specifying relationship (along with the corresponding generalizing relationship in the opposite direction). This same process is also possible with the high level relationships. When a designer finds a problem claim that is appropriate, but does not find a suitable corresponding design claim, he or she will have to create their own design claim. Creating a design claim and keeping the postulation/predication relationship in mind allows the designer to tailor the writing of the claim and assign that relationship when adding the new claim into the library. This is the partial reuse of a claim for the purposes of creating new claims with the ultimate goal of incorporating the most appropriate claim possible into a design. This process continues to solve our third problem of adapting claims to one's needs.

How can others reuse this new claim? Typically, adding a new claim to the claims library means it is isolated from the rest of the claims. The only way to find a new claim is to formulate a query that will return the claim as a result. The use of a relationship to create a claim explicitly allows the designer to acknowledge how the new claim can be included in the library. The assignment of this relationship between the base claim and the new claim prevents isolation, allowing other designers in the future to find this new claim (through selection).

This application of the claim relationships, in conjunction with the previous applications, completes the set of processes necessary to establish a cycle of reuse. The relationships are first used to select claims. If a claim is not found, a claim is reused to generate a new claim with a relationship. The claim is then added into the library's network of claims, thereby facilitating future reuse through the relationships. This cycle eventually contributes to the growth of the claims library and the identification of new design knowledge.

Understanding the implications of this process requires investigating how claims are created using a relationship and what relationship is chosen to create the claim. The creation of claims represents the molding of new knowledge and the choice of relationship used to create the claim is critical to how the claim can be found through selection. We present two studies that explore both these aspects.

Claim Creation

Having a standard claim creation process can greatly benefit designers. It can define what should be done to create a claim using a particular relationship, ensuring a higher

quality of reusable claims. Our intention was to investigate the possibility of an emerging standard claim creation process using relationships. As a result of this, we identified four trends in creating claims for each of the low level relationships. Use of the generalization/specification relationship to create a claim should preserve the underlying concept of the original claim, but have new upsides and downsides related to the claim feature. The fusion relationship should result in the new claim having all the tradeoffs of the base claims in a synthesized form, but a new feature. The diffusion relationship should only retain some of the applicable tradeoffs. Using the mitigation relationship should result in a new claim with a feature describing the solution to a particular downside in the base claim. Translating a claim should yield a claim with a similar feature with some tradeoffs being preserved.

We gave a set of 10 claims (some are in the explanations of the relationship types) representing the design of a notification system to 78 undergraduate HCI students. The students were asked to choose any two claims and create two more claims based off of the chosen claims. Half the students were asked to use the generalization/specification and translation relationships to create the new claims. The other half was asked to use the fusion/diffusion and mitigation relationships. To aid them, they were provided with written definitions and examples of each relationship type.

The use of the postulation/predication and execution/evaluation relationships for this study was avoided because it would force the students to rethink the design represented by the set of claims. Nevertheless, the study demonstrates how the low-level relationships can be used to create new claims.

Results and Discussion

Analysis of the created claims showed widely varying methods of creating claims. We were not able to conclude the existence of any distinct claim creation process for any of the relationships with absolute certainty. Hence, we can not accept any of the hypotheses we had. However, we can report on what the majority of students tended to do while creating new claims and note on similarities between what they did and what we anticipated.

For the generalization relationship students identified words or concepts that could be described in more general terms. For example, students would generalize red by referring to it as color. When specifying a claim, students tended to add adjectives to make certain conditions more specific. For both the relationships, the generalization and specification of the title were imperative. New upsides and downsides were created for both these relationships, but some were taken from the original claim if they still applied. Although we anticipated the creation of new upsides and downsides, we did not expect as many adoptions of original tradeoffs.

During the fusion process, a new claim encompassing all the claims being fused was created. The title usually contained concepts from all the claims being fused. Most of the students took the definition of fusion literally, combining the upsides and downsides from all the claims into one claim. When diffusing, students identified a single concept

and usually created a completely new claim. The fusion process was consistent with the behavior we predicted.

Mitigation and translation both resulted in the creation of a completely new claim with new upsides and downsides. While creating a mitigating claim, students made sure the title described a solution to the chosen downside. Translating claims were created by first identifying a word or concept in the original claim and then identifying an alternative concept for the new claim. The importance placed in the formulation of the title when mitigating was expected by us. This demonstrates the value placed in the title of claims themselves as another form of a succinct description. We did not foresee the creation of completely new translating claims. We believed many of the new translating claims would share many similarities due to the fact the relationship connects alternative claims to each other. Instead, the students took the chance to create alternative claims that were of a completely different nature.

This study demonstrates designers can in fact create new claims using relationships, proving valuable insight needed for the formalization of claim creation in the cycle of reuse. Each relationship is seen as a guide to writing a new claim. The partial reuse of claims is seen as an important catalyst for creation. There is a general theme of breaking down claims into concepts to create new claims. These concepts are identified through nouns, verbs, and adjectives that form the grammatical structure of the claims. This process reiterates the previously mentioned need for understanding how concepts within claims are identified. Currently this process is implicit, but it must be formalized and made explicit to preserve quality in new claims. An equally important concern in maintaining the quality of claim in the claims library is the validity of the relationships being used to create the claim. Our next study explores this aspect of the cycle of reuse.

Relationship Identification

The biggest hurdle to achieving the goal of future reuse is validity of the relationship a claim has. When creating a new claim using a relationship, how does one know they are using the correct relationship? Improper use of a relationship may mean that newly specified claims may not be found in the library when they are connected to other claims. Designers may not be looking for a claim with the improper relationship or they may lose confidence in the claims library when they see claims that do not have the chosen relationship. This problem is critical to how claims are specified using relationships and, as this can affect the browsing of claims using relationships, has implications for selection.

To investigate the process of relationship identification, we asked the 78 HCI students who did the claim creation activity to identify relationships between claims. The students divided into groups of three to four and were asked to identify as many relationships as possible among the claims in the previously given set of 10 claims. Definitions of each relationship and examples of their use were also provided.

During this process we wanted to gauge how well students could understand the definitions of the claim relationships. We anticipated the students would find the same

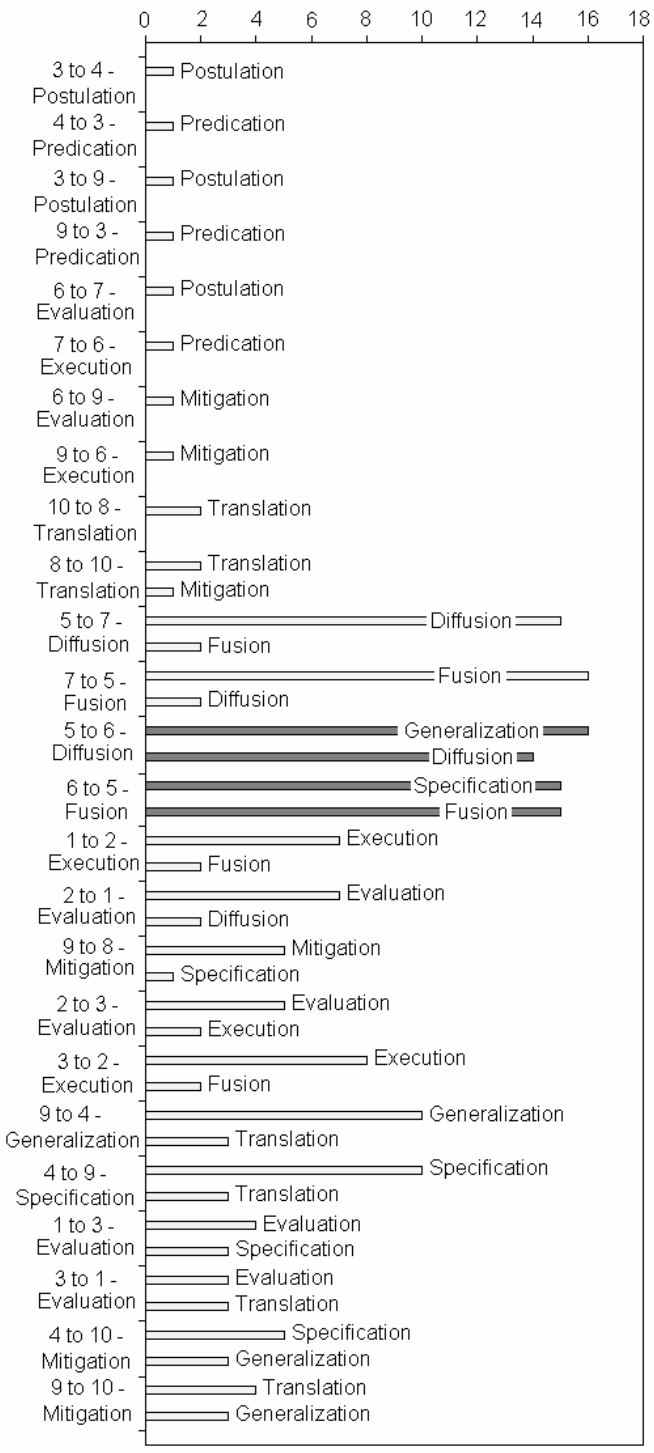

Figure 16. The 14 pairs that had relationships between the claims. The correct relationship type is given in the axis. For each pair, the highest and second highest assigned relationship is shown.

relationships we found among the 10 claims. We also expected to observe some degree of confusion between certain relationships although we were not sure of how widespread this condition may be and which relationships would lead to such cases. The analysis of the results proved to be eye-opening.

Results and Discussion

Our analysis was steered by specific points of investigation. These concerns are points critical to the sustenance of a relationships-based claims library with little or no moderation. If the claims library is to grow over time, it is imperative that the use of relationships does not undermine the utility of the library. Concerns must be alleviated if they prove to be a potential hindrance to the use of the claims library.

Point 1: Growing number of identified relationships. Our first concern is the identification of new relationships. A growing claims library must support the addition of these new relationships to expand utility. New relationships can arise as claims are used in different contexts over time. This raises the question of the semantic validity of assigned relationships.

In the given claim set we thought we had placed 20 relationships in 10 unique pairs. Students found 148 relationships in 57 unique pairs. There are two explanations for this occurrence: 1) there are relationships that were not discovered by us and 2) there are relationships which are not semantically valid.

The fact that there are undiscovered relationships within a set of claims rejects our first hypothesis and supports the need for continuous additions to the claims library. Indeed when we looked back at the results, there were claims that could actually have the relationships the students found. After careful examination of the results, we accepted that there were actually 26 relationships in 14 unique pairs. Figure 16 shows the 14 pairs along with the correct relationships and the students' assignments. (We count the translation relationship as one relationship because of its bidirectional nature. We also accept that claims 5 and 6 have two different relationships between them. This brings our total count to 26 relationships.)

However, there are many assigned relationships that are semantically incorrect. Referring to Table 3, we see about 50% of the pairs of claims had 2 or more different relationship types assigned. Note that pairs having 5 or 6 different relationships types assigned had the most total number of relationships assigned. The results show that there is a concentration of pairs that have high disagreement in relationship assignment, predicted by our second hypothesis. This indicates that certain pairs were definitely points of extended discussion among the students.

We explain high disagreement among a concentration of pairs as a result of students knowing that a relationship exists, but not knowing the exact nature of the relationship, a common problem in the assignment process. Certainly the amount of disagreement must be decreased. This must be done by guidance provided by the claims library. Library users must be taken through a guided process of identifying a certain relationship based on two claims. The definition of this process will require further research.

Number of different relationship types assigned to pair	Number of pairs having the number of different types of relationships	Number of unique relationships assigned to these pairs	Total number of relationships assigned to these pairs
1	29 (50.8%)	29 (19.5%)	41 (9.0%)
2	6 (10.5%)	12 (8.1%)	46 (10.1%)
3	5 (8.7%)	15 (10.1%)	18 (3.9%)
4	5 (8.7%)	20 (13.5%)	40 (8.8%)
5	4 (7.0%)	20 (13.5%)	81 (17.8%)
6	5 (8.7%)	30 (20.2%)	178 (39.2%)
7	2 (3.5%)	14 (9.4%)	33 (7.2%)
8	1 (1.7%)	8 (5.4%)	16 (3.5%)
	Total number of pairs that have a relationship: 57	Total number of unique relationships found: 148	Total number of relationships assigned to all pairs: 453

Table 3. Results acquired from the relationship identification study. The results show that undiscovered relationships exist, but that many discrepancies in the nature of a relationship exist.

Point 2: Identification of relationships in the opposite direction. Our next concern was to find out whether the students were able to identify relationships in the opposite

direction, assuming a relationship in one direction was identified. This analysis was geared towards syntactic validation of the use of relationships.

When comparing the results of all the possible pairs, we observed differences among the relationships types. The postulation/predication and generalization/specification relationships had no significant problems, showing the students accurately understood the use of these relationships. The translation relationship is bidirectional and therefore did not cause any disparities either. The mitigation relationship, however, is a unidirectional relationship. Despite this fact about half the claims were assigned the relationship in both directions. Many students were not clear on whether a relationship in the opposite direction should exist when using the execution and evaluation relationships.

Syntactic errors should be the first types of errors to be eliminated. This analysis shows that we must focus on creating better definitions for each relationship type, particularly the execution/evaluation and mitigation relationships.

Point 3: Similarities between relationships. Zimmer (1995) explained how the assigning of relationships can be difficult due to confusions about which type is more appropriate. Similarities between relationships are important when understanding the use of relationships in the claims library. They represent the (lack of) uniqueness of a single relationship type and how much confusion can be caused when assigning the type to claims. We found two examples of similarities and confirm our second hypothesis regarding confusion between relationship types.

The best example of a similarity is found between the generalization/specification and fusion/diffusion relationships—apparent in claims 5 and 6 in Figure 16. This pair was the best pair of claims that represented the fusion/diffusion relationship and had 15 fusion and 14 diffusion relationships assigned in opposite directions. The same pair also had 16 generalization and 15 specification relationships assigned. This points us toward an interesting phenomenon. Closer investigation shows that the result of fusion of two or more claims results in a specification of all claims. Naturally, the result of diffusion becomes a generalization of the claims. This shows that the generalization/specification relationship is a more basic relationship that can be built upon. Problems may arise when assigning relationships if two relationship types seem to describe the nature of the relationship between claims. We suggest the need for a prioritization of relationships in an assignment process to avoid having more than one relationship type being assigned.

Another similarity exists between the generalization/specification and translation relationships. There were 42 and 44 instances of generalization and specification relationships respectively across 13 unique claim pairs. 7 of these pairs also had a total of 18 translation relationships, showing the second largest pattern between two different relationships types. We find the only difference between these two relationships is the role of scope. In the generalization/specification relationship, two claims regarding the same basic idea are related due to one being for a broad scope and the other being for a narrow scope. In the translation relationship, two claims are related because they have the same scope and are possible alternatives to each other. Confusion can arise when two claims are alternatives to each other, but have different scopes. For example, a claim

about a blue blinking button and another about a green button are alternatives to each other because of the colors. However, the blue blinking button claim is slightly more specific in scope than the other claim. The relationship between these two claims should be translation because they are alternatives based on the underlying concept of color. We suggest the need for some form of guidance provided by the claims library itself to avoid such cases. The guidance should allow users to identify underlying concepts within claims and identify the most suitable relationship.

Both examples of similarities illustrate the need for prioritization and concept identification. When two different relationship types seem to be appropriate, a priority should be enacted so consistently choose the same relationship type. With the ability to identify concepts, confusion caused by problems related to scope and be overcome.

This study has allowed us to critically analyze the use of the relationships to judge their success in the claims library. We have found that there is much work that remains to be done with the process of assigning relationships. Both this study and the claim creation study motivate the need for tool support to aid the generation of claims and identification of relationships. This tool will require better definitions for each relationship type, prioritization for superceding relationship types, and claim concept identification and comparison. Success with implementing these measures should decrease the amount of disagreement resulting from syntactic and semantic errors seen in relationship identification and maintain the quality of new claims.

The next section discusses how claims can be integrated once they are gathered through the reuse aspects demonstrated thus far. A method of connecting claims together to represent a design is presented. Implications of these connections are outlined in a study.

5.4. Integration

So far we have described how one can identify the correct type of claim through abstraction, browse through a network of claims to select, and specify new claims to adapt. We must now discuss how the claims can be connected together by the designer—an issue addressed by the last reuse aspect: integration. Integration is the process in which a collection of selected and specialized components are combined to form a system (Krueger, 1992). As part of the creative design process, designers using the claims library will most likely have claims that have never been used together before. The aggregate of all the claims they collect will somehow represent the design they want to create. All the claims must be integrated into a design such that every claim's role within the context of the whole design is understood. We must also be able to understand what the final result of this process means. A designer should gain value from not just integrating single claims, but from the combined state of all the claims.

Laurian has finally collected and created many claims representing various aspects of the notification system she wants to produce. She now wants to bring together all the knowledge to show how different ideas will work together in her design. She combines features of the system together to make sure the requirements are met in the design. As she does this, she notices certain design features have negative effects that need to be accounted for. As a result, she gains insight into future directions for her design work. She continues to iteratively design her system and add new ideas when necessary.

Figure 17. A scenario showing the need to integrate components that will be reused.

How these claims should come together and interact with each other is the final challenge the designer must face. Without an overall understanding of the final conceptual design, designers will find it harder to agree on the most important aspects of the system. We propose the final application of claim relationships be a method for integrating claims toward providing an enduring record of design decisions reflecting how and why claims were integrated in the design process.

Two claims can be integrated with each other when a relationship is assigned to connect them. This behavior can be continued until a group of claims are linked to each other. The structure of claims and the nature of claim relationships allow us to uniquely identify benefits of integration. First, the subsequent group of claims permits a designer to outline high level concepts and strengths and weaknesses of the design being worked on. The collection of claims formed as a result of selection and specification may contain general claims describing overall goals and specific claims which describe system features. When integrated, the goals are explicitly associated through a relationship to the features that enact them. Advantages and disadvantages are expressed in the tradeoffs of the claims and the relationships used to connect them. For example, mitigation shows a solution to an identified problem. The second benefit is the ability for such integration to demonstrate opportunities for testing and redesign. When viewing the integrated claims one can gain an understanding of what is lacking. They indicate when solutions to certain weaknesses do not exist through the absence of certain integrations. This holistic view may even cause designers to determine groups of claims to be insufficient—a possible catalyst to an analytic study by experts. The third gain in integrating claims is the ability for them to represent a trace of motivating factors for iterative improvements over time. Over multiple iterations more claims may be selected or specified and then integrated with preexisting claims.

We refer to a group of integrated claims as a *claims map*. Claims maps are directed graphs of claims showing how all the claims collected for a design will work together, representing a decomposition of the whole design. The claims are connected to each other using the claim relationships. Although an explicit hierarchy is not necessary, it is possible for a designer to choose to do so. We believe claims maps are a gateway to portraying the three benefits of integration and act as a tool for designers to negotiate design concerns.

Claims Map Creation

To explore the concept of claims maps, we simulated a development session in which 18 graduate students were asked to examine design decisions made in a published paper regarding a system toward understanding how the original designers captured claims relationships and how an explicit representation of them can help future redesign. Each student was first asked to individually create their own claim based on an aspect of SideShow (Cadiz et al., 2002). SideShow is a notification system that docks to the side of the screen to provide updates on information such as weather conditions, number of bugs, and online buddies. We asked the students to create claims individually so we could mimic a real design process where a designer would gather different claims and try to integrate them with each other. After this initial step, the students were then divided into four groups, with 4-5 students in each group, and asked to create claims maps to describe the system as a whole in terms of claims.

Results and Discussion

Our method of asking participants to create claims individually proved successful because each group had a diverse set of claims which described different aspects of the same system. The new claims varied widely among all the students. Some claims concentrated on specific features of the system while others encapsulated general design goals of the system. Because students were taking different claims and trying to integrate them, we expected the students would have to create new claims. Indeed, the students did create new claims to bring other claims together. On average, each group created 4 more claims to have a claims map suitable enough to represent the system. We provide an example of one of the claims maps developed by a group which had four members (see Figure 18).

Students showed the first benefit of claims maps in the types of claims they used. General claims, such as the claim about management and organization of information in Figure 18, depicted high-level goals addressed by the system. They outline motivations for building the system. Specific claims describing the features were identified and integrated with the high-level claims, proposing solutions for the goals. The advantages and disadvantages of the system are encapsulated in the various tradeoffs of the claims. A mitigation relationship in the example claims map points out a specific weakness of the design that was addressed by another claim to strengthen the system. Through quick recognition of such design characteristics, one can notice the immediate utility of claims maps.

The second benefit can be seen within the opportunities created for testing and redesign. These chances are instantiated when new claims are created, downsides are left unaccounted for, and relationships need validation. Many of the concepts encapsulated by the claims were derived from the literature about the SideShow system, however, many others were created during the integration process, initiating a need for testing. The new claims and their tradeoffs represent aspects of the design that should be tested to validate the effects of the feature. For instance, the video feed claim in the example was created as a result of a specification to demonstrate the system's ability to deliver

dynamic information. This was not specifically addressed by the SideShow publication (Cadiz et al., 2002), but instead was an observation made by the students. Newly identified features and tradeoffs need to be validated in various forms. Certain claims may need small analytic studies while other groups of claims may require longer field studies. Further opportunities for redesign arise when downsides are unaccounted for (through the use of mitigation). Significant downsides can represent potential risks to the design of the system, requiring immediate attention. Determining the magnitude of certain downsides is a catalyst to conducting further testing. The validity of certain relationships can also be a point of contention if they are not evaluated. Is the claim in Figure 18 about using icons truly an alternative to providing short summaries? What is it about these two claims that make them alternatives to each other? Does the collapsible groupings claim really mitigate the problem of hidden information? These are pressing questions situated within the nature of the relationship used for integration that can only be answered through evaluations.

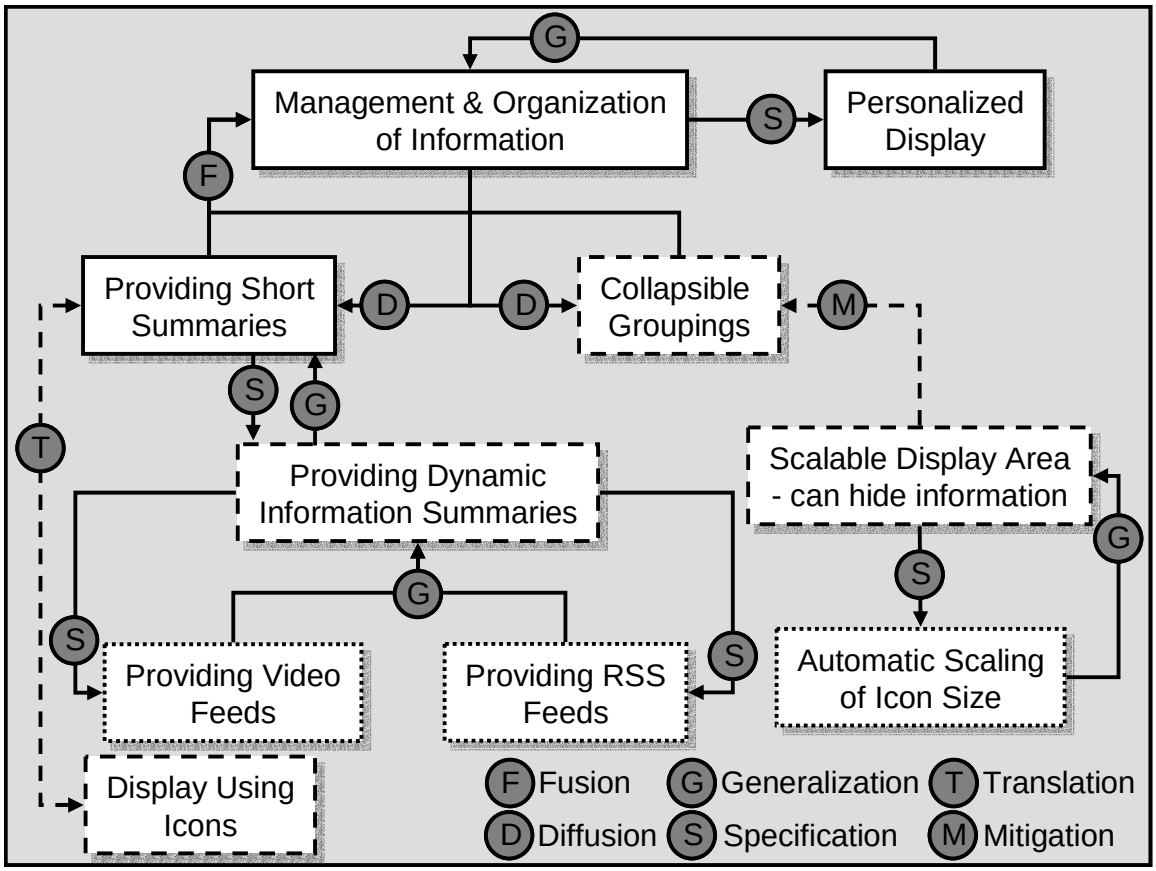

Figure 18. A claims map about the SideShow system created by one of the participating groups. Many claims are from literature (Cadiz et al., 2002) while others were derived using the relationships. Questions about the validity of certain claims and relationships (indicated by the dashed line) and claims not mentioned in the paper, but created by students (shown by the dotted lines), should be answered through testing.

The accumulation of opportunities for testing further motivates the use of claims maps as an evaluation tool. The map can be given to expert evaluators to interpret the

state of the design and determine where further redesign should occur. These observations situate claims maps as a tool to gauge the state of the design and trigger evaluations and redesign efforts when needed.

The first two advantages of using claims maps build to the emergence of the third benefit. Over a longer period of time continuous integrations create a trace of design activities. With each addition of a new claim, the claims map records evidence of a new iteration. Looking back at such a footprint allows designers to trace back to the origins of design intentions. Designers find the opportunity to investigate the history of a design and learn lessons from design decisions that were made.

We believe claims maps can establish a format for creating a full fledged design record with proper tool support, allowing designers to truly gain value from the third benefit. We envision a tool that tracks the development of a claims map over time, showing when claims were integrated. Design rationale explaining the additions or even the removal of claims can provide further insight into the development. Tracking of tested and untested claims will allow designers to monitor the state of evaluations. With the three benefits becoming more evident, it is to such a tool that we should strive to work toward.

6. CONCLUSIONS AND FUTURE WORK

Our goal of promoting reuse through the work we have presented has been based on two important concepts: Krueger's (1992) aspects of reuse and claim relationships. This work can serve as a proof of concept for how our discipline can begin to reuse knowledge. We are advocating the use of the reuse aspects as a possible guiding model to reuse in HCI. Equally important is that one must have an approach to implementing these reuse aspects—ours being based upon the use of claim relationships. They not only implement the reuse aspects, but also serve as a framework to support design. Using relationships for abstraction and selection allows designers to identify and retrieve the correct types of claims from a repository. Claims creation through specification supports designers who need to tailor claims for their own work. Finally, claims maps force designers to integrate claims and consider the low level details of how their design should really function.

This work provides three main contributions to those who wish to explore the methods for how to reuse and design:

- Establishing the need for reuse within HCI and providing an example of how a software engineering perspective is relevant to the problems we face in HCI regarding the reuse of design knowledge.

- Presenting an approach to reuse based on claim relationships to enact Krueger's vision of reuse.

- Laying the groundwork for how claim relationships can be applied in different ways to enact each reuse aspect and support designers.

Of course our work is not complete. We must conduct a broader and complete study of reuse in HCI. The use of claim relationships for abstraction, selection, specification, and integration should be investigated together, putting emphasis on connections between each reuse aspect. This will certainly have to be done using larger design activities over longer periods of time, allowing us to track and analyze the usage of not just the high level relationships, but also the low level relationships. The relationships have the potential to affect the quality of designs created in SBD. This facet must certainly be investigated.

Our work regarding the overall use of the claims library is not complete either. We strive to create a claims library that will not be centrally-controlled, but instead will be community-controlled. Such environments can become chaotic without the proper tools and techniques to maintain critical resources and processes. Since a lack of control can lead to an environment that may eventually be rendered useless, we must begin to understand the types of support that will be needed to maintain the quality of the library contents. With respect to claim relationships, the assignment of relationships may become a very inconsistent process, leading to networks of claims that lack consistency. Tools that will better define relationships and aid in their assignment will be needed for community members. Another concern is the quality of new claims that are added to the library. Claims that are generated using relationships may not be written in a reusable style, making many of the claims hard to reuse. We must support explicit processes for creating claims based on relationships to ensure new claims are created with speed, efficiency, and, most importantly, quality.

Taking a step back, we must think about the nature of the claims library itself. We should think of what it means to design a digital library. Krueger demonstrates what is necessary to have a viable reuse approach, but Goncalves et al. (2004) point out that there are many concerns that need to be taken into account when designing a digital library. The digital library community can certainly help with the methodology required to build such a library. The 5S framework describes streams, structures, spaces, scenarios, and societies as the most important aspects when designing digital libraries (Gonçalves et al., 2004). Streams represent the types of data stored in a library. Structures outline how parts of the whole are organized. A space describes the space in which objects exist and are manipulated based on a set of operations. Scenarios illustrate system behavior form the user's point of view. Finally, societies consist of the humans, hardware, and software and their inter-relationships with respect to the digital library. With the work presented here, we increasingly begin to see the importance of the 5S model within the context of our claims library. Claims are comparable to the streams described by the model. The claim relationships and consequential cluster of claims provide structure. Spaces in which claims are retrieved from the library and manipulated based on the outlined tools must exist. Scenarios must outline the explicit steps users will go through when searching for claims and using the relationship identification and claim creation tools. Finally, the HCI students in our undergraduate and graduate courses form the target society. With these connections in mind, our goal is to continue to improve and develop the claims library using the 5S model.

REFERENCES

Alexander, C., Isikawa, S., & Silverstein, M. (1997) *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press.

Basili, V. R., Briand, L. C., & Melo, W. L. (1996) How reuse influences productivity in object-oriented systems. *Communications of the ACM*, 39, 10, 104-116

Biggerstaff , T. J. & Perlis, A. J. (1989a) *Software Reusability: Vol. 1, Concepts and Models*. New York: ACM Press.

Biggerstaff , T. J. & Perlis, A. J. (1989b) *Software Reusability: Vol. 2, Applications and Experience*. New York: ACM Press.

Blanford, A., Stelmaszewska, H., & Bryan-Kinns, N. (2001) Use of Multiple Digital Libraries: A Case Study. *In 1st ACM/IEEE-CS Joint Conference on Digital Libraries*.

Borchers, J. O. (2000) A pattern approach to interaction design. *In Proceedings of the conference on Designing interactive systems*, 369-378

Cadiz, J. J., Venolia, G., Jancke, G., & Gupta, A. (2002) Designing and Deploying an Information Awareness Interface. *In Proceedings of Conference on Computer Supported Cooperative Work (CSCW '02)*, 314-323

Carroll, J. M. & Kellogg, W. A. (1989) Artifact as theory-nexus: Hermeneutics meets theory-based design. *In Proceedings of the Conference on Human Factors in Computing Systems (CHI'89)*, 7-14.

Carroll, J. M. & Rosson, M. B. (1992) Getting Around the Task-Artifact Cycle: How to Make Claims and Design by Scenario. *ACM Transactions on Information Systems (TOIS)*, 10, 2, 181-212.

Carroll, J. M. (1994) Making use: a design representation. *Communications of the ACM*, 37, 12.

Carroll, J. M., Kellogg, W. A., & Rosson, M. B. (1991) *The task-artifact cycle*. Designing Interaction: Psychology at the Human-Computer Interface. (pp. 74-102) Cambridge Press.

Carroll, J. M., Singley, M. K., & Rosson, M. B. (1992) Integrating Theory Development with Design Evaluation. *Behavior and Information Technology*, 11, 247-255.

Chewar, C. M., Bachetti, E., McCrickard, D. S., & Booker, J. (2004) Automating a Design Reuse Facility with Critical Parameters: Lessons Learned in Developing the LINK-UP System. In *Proceedings of the International Conference on Computer-Aided Design of User Interfaces (CADUI '04)*, 236-247.

Chewar, C. M., McCrickard, D. S., & Sutcliffe, A. G. (2004). Unpacking Critical Parameters for Interface Design: Evaluating Notification Systems with the IRC Framework. *In Proceedings of the Conference on Designing Interactive Systems (DIS '04)*, 279-288.

Creech, M. L., Freeze, D. F., & Griss, M. L. (1991) Using Hypertext in Selecting Reusable Software Components. *In Proceedings of 3rd Annual ACM Conference on Hypertext and Hypermedia*, 25-38.

Dusink, L., & van Katwijk, J. Reuse Dimensions. (1995) *In Proceedings of Symposium on Software Reusability*, 137-149.

Embley, D. W. & Woodfield, S. N. (1987) A Knowledge Structure for Reusing Abstract Data Types. *In Proceedings of 9th International Conference on Software Engineering*, 360-368.

Gall, H., Jazayeri M., & Klosch, R. (1995) Research directions in software reuse: where to go from here? *In Proceedings of Symposium on Software Reusability*, 225-228.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995) *Design Patterns: elements of reusable object-oriented software*. Boston: Addison-Wesley Longman

Godin, R., Pichet, C., & Gecsei, J. (1989) Design of a browsing interface for information retrieval. *Proceedings of the 12th Annual International ACM SIGIR conference on Research and development in information retrieval (SIGIR '89)*, 32-29

Gonçalves, M., Fox E., Watson, L., & Kipp, N. (2004) Streams, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Libraries. *In ACM Transactions on Information Systems*, 22, 2, 270-312.

Greenberg, S. & Rounding, M. (2001) The Notification Collage: Posting Information to Public and Personal Displays. *Proceedings of the SIGCHI conference on Human factors in computing systems* pp 524-521

Henninger, S. (1997) An evolutionary approach to constructing effective software reuse repositories. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 6, 2, 111-140

Horowitz, E. & Munson, J. B. (1989) An expansive view of reusable software, In T. J. Biggerstaff & A. J. Perlis (Eds.), *Software Reusability: Vol. 1, Concepts and Models* (pp 19-41). New York: ACM Press.

Krueger, C. W. (1992) Software Reuse. *ACM Computing Surveys (CSUR)*,24, 2, 131-183.

Landay, J. A. & Borriello, G. (2003) Design Patterns for Ubiquitous Computing, *Computer*, 36, 8, 93-95

Maiden, N. A. M. & Sutcliffe, A.G. (1994) Requirements Critiquing Using Domain Abstractions. *In Proceedings of IEEE Conference on Requirements Engineering*, 184-193.

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

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

Payne, C., Allgood, C. F., Chewar, C. M., Holbrook, C., & McCrickard, D. S. (2003) Generalizing Interface Design Knowledge: Lessons Learned from Developing a Claims Library. *In Proceedings of the IEEE International Conference on Information Reuse and Integration (IRI '03)*, 362-369.

Prieto-Diaz, R. (1989) Classification of reusable modules. In T. J. Biggerstaff & A. J. Perlis (Eds.), *Software Reusability: Vol. 1, Concepts and Models* (pp. 99-123). New York: ACM Press.

Ralph E. Johnson, Frameworks = (components + patterns), *Communications of the ACM*, 40, 10, 39-42

Rosson, M. B. & Carroll, J. M. (2002) *Usability Engineering: Scenario-Based Development of Human Computer Interaction*. Morgan Kaufmann Publishers.

Sugumaran, V., Tanniru, M., & Storey, V. C. (2000) Supporting reuse in systems analysis, *Communications of the ACM*, 43, 11es

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

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

Wagner, W. P. (1990) Issues in Knowledge Acquisition. *In Proceedings of ACM SIGBDP Conference on Trends and Directions in Expert Systems*, 247-261.

Wahid, S. Allgood, C. F., Chewar, C. M., & McCrickard, D. S. (2004a) Entering the Heart of Design: Relationships for Tracing Claims Evolution. *Conference on Software Engineering and Knowledge Engineering (SEKE '04)*, 167-172.

Wahid, S., Smith, J. L., Berry, B., Chewar, C. M., & McCrickard, D. S., (2004b) Visualization of Design Knowledge Component Relationships to Facilitate Reuse. *In Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration (IRI '04)*, 414-419.

Whittaker, S., Terveen, L., & Nardi, B. A. (2000) Let's stop pushing the envelope and start addressing it: A reference task agenda for HCI. *Human-Computer Interaction*, 15, 75-106.

Ye, Y. & Fischer, G. (2002) Supporting reuse by delivering task-relevant and personalized information. *In Proceedings of Conference on Software engineering*, 19-25

Zimmer, W. (1995) Relationships Between Design Patterns. In J. O. Coplien and D. C. Schmidt (Eds.), *Pattern Languages of Program Design* (pp. 354-364). Addison-Wesley