Designing the Claims Reuse Library: Validating Classification Methods for Notification Systems

Alain Fabian, David Felton, Melissa Grant, Cyril Montabert, Kevin Pious, Nima Rashidi, Anderson Ray Tarpley III, Nicholas Taylor, C. M. Chewar, and D. Scott McCrickard

Center for Human Computer Interaction and Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA USA 24061-0106

{alfabian, dfelton, megrant, cmontabe, kpious, nrashidi, atarpley, nitaylo1} @ vt.edu, {cchewar, mccricks} @ cs.vt.edu

ABSTRACT

We discuss our research into the development and testing of a notification system claims library for assisting designers in interface development. Our research focuses on achieving consistent values among multiple users when adding and searching claims. We discuss the methods used for redesigning the application, techniques used for testing, and reengineering goals for the Claims Library. This work extends previous efforts on design knowledge reuse in the HCI research community, as such our methods and techniques should be reusable by others. We designed the interface to the library for users entering claims, ensuring usability and understandability. Since we noted problems with a particular feature (the IRC input method) through an internal round of testing, we conducted a lab-based test to isolate specific breakdowns. Our results validated portions of claim classification indices, suggest key reengineering changes that should inform ongoing and future development of the claims library—of broad interest of notification systems developers.

Categories and Subject Descriptors

H.1.2 **[Models and Principles]**: User/Machine Systems - *Human Factors*; H.1.1 **[Models and Principles]**: Systems and Information Theory – *General systems theory*

General Terms

Performance, Design, Experimentation, Human Factors

Keywords

Notification system, claim, reuse, user testing, interface design

1. INTRODUCTION

Notification systems are used to inform users of valuable information in an efficient manner without unnecessary interruption to their primary task. The primary task is the main focus of the user's attention. The notification task is accessing digital information from sources secondary to current activities. There must be a dual task aspect established where the balance between the primary task and the notification task is optimal for productivity. To obtain maximum productivity, there must be an ideal level of interruption from the primary task, reaction to the notification sent, and comprehension of information presented by the notification [1]. Monitors such as Norton Antivirus™, tickers such as ESPN BottomLine, and software helpers like the Microsoft® Office® paperclip inform users without pulling them away from their primary task. From a human computer interaction perspective, designing an effective notification system is a difficult task.

How could we make designing a notification system easier? Each notification system begins with a design problem that focuses on the primary and notification tasks. Designers must consider the level of interruption that is acceptable, the amount of screen space used if the system is computer based, the level of configurability, use of colors and sounds, and other important factors before implementation can begin. Through a knowledge repository we intended to make designing such interfaces simpler because of the reusability of the stored ideas.

Successfully designing a repository would allow designers to retrieve ideas that would broaden their knowledge and expand the known solutions to solve their problems. This should enable them to produce a higher quality interface. Through their design and testing, they will discover new ideas that they will be able to add to the repository. This approach has been argued in HCI literature [2, 3, 5], although it has not been proven by a working system. This cycle of using and adding ideas would make the repository stronger and more valuable to users.

2. BACKGROUND

To fully understand the value of this repository the user must have access to the information stored. A broad range of ideas will be available to users, but each is structured in a way to make them understandable and applicable in more than one way. Two useful concepts on which we built our research are the ideas of a claim and a claim library.

2.1 What is a claim?

A *claim* is a statement that describes the effect a feature will have on a user within a usage scenario [3]. The ideas presented in a claim are based on empirical testing or observations made which allow designers to compare one claim with another based on design techniques, testing outcomes, or numerous other fields. Claims make explicit the ideas that are present in designing a notification system.

ACMSE '04, April 2–3, 2004, Hunstville, Alabama, USA.

Claims consist of a title, description, upsides and downsides that accompany the description, scenarios, theories and artifacts that support the description [3, 4]. The IRC framework [1] is used to describe the desired levels of interruption, reaction, and comprehension brought about by the specific feature (Table 1). Matching the feature's actual IRC level with an intended IRC level is thought to enhance the usability of a notification system.

2.2 What is a claims library?

A *claims library* is a repository that stores claims that are reusable and can make notification system design easier and quicker for designers. This impact on difficulty and speed is a result of the reuse of claims being a "powerful form of knowledge transfer" [2]. Designers will be able to search on multiple criteria and receive matching claims as results. When a quality claim is found and used in the design, less empirical testing will be needed, which will allow rapid prototyping. Having a claim to base their product on will allow developers to avoid certain mistakes and ensure they include important functionality. Development time is always an issue and ideally, designers value the ability to narrow the problem specification in order to create a system based on a set of requirements on the first attempt.

3. RESEARCH PROBLEM

In this study we investigate how consistently people specify and select six main attributes of the claim: the interruption, reaction and comprehension values as well as the notification tasks, design concerns, and primary notification environment that accompany a claim in the design process. Resolving these issues is essential in the creation of a claim specifying and searching interface.

3.1 Problem Approach

We approached this study in four phases: initial prototyping, pilot testing, formal lab-based testing, and analysis of results. Initial prototyping activities are discussed in section 4. Section 5 discusses our pilot testing and formal lab-based testing procedures and results. Following the review of these four phases, we discuss general findings, conclusions, and directions for future work in section 6.

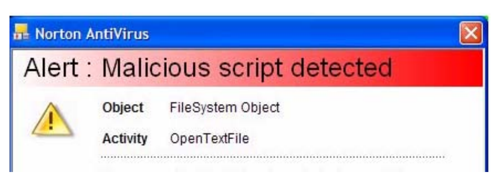
4. INITIAL PROTOTYPING

Our first phase consisted of initial prototyping activities to create a claims entry and editing form. These activities focused on six different features that supported consistent and high quality claim entry: multiple entry pages, explanation of entry fields, user claim rating, supportive media, checklists for specifying aspects of feature usage, and IRC values.

Brainstorming sessions were conducted to discuss the best way to implement each of these features. We wanted to ensure that the interface enabled any HCI student to enter a claim in a consistent fashion. We worked under the assumption that students would have no previous knowledge of the claim attributes or how to decide which values best describe the claim (such as IRC values).

Throughout adding new features to the system, we tested the effectiveness of our interface by adding claims to the library ourselves. This allowed us to experience the strengths and weaknesses first hand so we could understand where our users would have problems. Each of the features that we prototyped are described below.

Table 1. Example claim

Title	Virus monitoring through informational popups
Description	Virus scanning is done through the background, and when problematic and possibly infected files are found a popup will appear (regardless of what program currently has focus) and will prompt the user to delete, quarantine, or ignore the virus that was detected. The Symantec™ name of the virus is also displayed.
Upsides	+ Quick notification to the user of the virus activity regardless of what program has window focus. + Brief description of the virus (the name) allows user to research on his own on how to react. + Three commonly selected options allow the user immediately to react and save his system.
Downsides	- Minimal information is displayed, leaving virus information gathering up to the user. - Advanced options (such as using specific removal tools) aren't available from the brief popup. - Popups when the virus is easily seen (such as on emails that obviously shouldn't be opened) can become annoying.
Design Issues	Is a more descriptive block of information on the virus warranted or desirable? Would more options on reacting to the virus be detrimental to the simple interface of the popup window? Should a configurable sound alert be added in case the user is away from his computer, or running an application that won't allow the popup to take focus?
Scenarios	Sarah is working on a spreadsheet to keep track of her balances. She has been working for hours, and decides to take a much-needed break to check her email. She only sees one new message from her friend Lisa, containing a brief message with a small executable attachment. Sarah clicks on the executable attachment, watches a brief fireworks animation as a result, then deletes the email and returns to her spreadsheet program. After a few minutes of working on her balance sheet, her virus program pops up a window notifying her of a new Worm virus that was installed on her system. She then presses the quarantine button and returns to her spreadsheet work.
Artifact	Norton Antivirus™ 
IRC	Interruption: 1 Reaction: 1 Comprehension: .5

4.1 Multiple Entry Pages

In the design of the claims entry page, we decided to have the form in a series of pages. This would allow the user to see their progress through a status bar and would avoid the likelihood that they get overwhelmed by one long entry page. We grouped the fields into sections and used these sections as the four page divisions. This allowed us to have two to four entry requirements per page, which achieved a balanced ratio between information and the total number of pages. A fifth page was added for confirmation of claim information. This enabled changes to be made before the claim is added to the database.

4.2 Explanation of Entry Fields

Knowing what to place in each field of the claim entry form requires a high level of HCI background. For every field we added a brief explanation of what is needed for that particular entry. This would help a user by giving them an idea of what is expected. Under each description, we added a link that would show examples for the fields of the claim. This would assist the users if they require more explanation of what type of information is expected. The link opens a small pop-up window showing a sample of what type of information goes in the field. In this window, we used screen shots of existing systems with high, medium, and low ratings to assist in the explanation. Part of our user testing focused on how effectively the explanations guided the user's claim entry.

4.3 User Claim Rating

With the purpose of a claims library being to assist users in the creation of notification systems, we wanted to support user feedback on claim effectiveness. We created a rating system based on a point scale that is calculated by summing the values of six variables. The points for each variable are tallied and the number of stars is determined based on the number of points. Zero to nine receives one star, ten to nineteen receives two stars (Fig. 1), twenty to twenty nine receives three stars, thirty to thirty nine receives four stars, and forty or more points receive the full rating of five stars. There are no fractional star values for simplicity.

Figure 1. An example of the star results.

The variables used to calculate the rating are:

- *Theory:* If the claim has a theory connected to it, it is worth between five to ten points depending on the quality of the theory. If there is no theory then it is worth zero points.
- *Ratings from other claims by user:* The average of all of the user's previous claims' ratings divided by six since there are six variables. For instance if the users average rating is a 25 then this claim would get four points (Point totals are rounded).
- *Author's Experience:* Each author would have one of three experience ratings. Beginners receive no points, novices receive five, whereas experienced users receive ten.
- *Artifact:* If the claim has an artifact then it gains five points.
- *Number of hits to claim:* The number of times the claim has been viewed, the more hit points the claim earns. There is a scale assigning zero to ten points based on the number of hits.
- *Other users' rating of the claim:* The average user rating would be divided by six. See "Ratings from other claims by user" for example.

Testing the comprehensibility of this rating system was a key objective of the user rating tests.

4.4 Supportive Media

Our view is that including supportive media will make the claims easier to understand and the artifact mentioned easier to visualize. To this end, we included an option to upload pictures or other forms of media that will support the use of the claim in the scenario. The media examples that are uploaded will appear below the scenario when the claim is viewed.

4.5 Checklists for Specifying Feature Usage

We wanted to create an effective way for users to specify the features and usage environment addressed by a claim. A checklist helped in selecting items related to the claim's tasks as well as the design concerns. On the form, there is a list of options for the user to select which describe Notification Environment(s), Notification Task(s) and Design Concern(s). The rationale behind the checklist was that it gave the user an idea of what features for which to look and helped to avoid possible problems that may arise, such as not obtaining a match due to misspellings or differences in terminology. By using the list, we expect users of the system to obtain more quality hits from the claims library.

4.6 IRC Values

We anticipated that the entry of the Interruption, Reaction, and Comprehension (IRC) values would be the most difficult part of achieving consistency. The members of our design group had different ways of judging items such as the level of interruption, even after producing a basic decimal scale chart on a range from 0.0 to 1.0 to help figure the value. Due to the inconsistency, we decided that an algorithm to determine the values would be beneficial [5]. We implemented a "Get Parameters" function that acts as a "wizard" to evaluate the answers to eight multiple-choice questions that focus on different aspects of the IRC and uses those as the users' entries for the IRC parameters.

5. TESTING, RESULTS, AND ANALYSIS

The second phase of our research consisted of validating the design choices presented above, which were implemented in prototype form. Multiple rounds of testing were administered with participants that were typical of those who would be entering claims. Between these rounds, changes were made to improve the answer consistency among the participants.

5.1 Preliminary Testing

Our primary objective for preliminary testing was to see if the initial interface design was effective for entering claims. We were interested in isolating features that needed further development. We tested six students who had successfully completed an HCI course offered at Virginia Tech. These users were given four claim entries that were missing the IRC values, primary tasks, notification tasks, and design concerns. The users were asked to fully review the claims and then decide on the IRC values and complete the primary tasks, notification tasks, and design concerns checklist. The participants used the web interface to enter their results. They also used the help windows available on the interface to guide their decisions.

The initial test of former HCI students did not bring about the results we were expecting. The results validated the rating options and the screen flow; however, the IRC ratings, primary tasks, notification tasks, and design concerns were not consistent. Participants were confused by the meanings of some of the terms. In particular, the "Primary Task" term was the most confusing because participants did not understand the difference between each option. Towards the end of the test, the participants were becoming frustrated. Perhaps another factor that added to their frustration was the claims that were chosen were from artifacts that the participants had little to no experience using. It was difficult for them to understand newly explained options with an unfamiliar claim.

5.2 Redesign and Internal Validation

As a result of the first round of testing, a second iteration of design was undertaken. In this redesign, we removed the "Primary Task" option and replaced it with "Primary Notification Environment" because we thought the category was more understandable for users to categorize claims. Although the list of generalized tasks we originally used for classification of primary tasks allowed cross domain reuse [4], we wanted to start with a less abstract classification approach. Additionally, we changed the IRC parameters to be high, medium, or low, instead of a decimal value between 0.0 and 1.0. We believed that giving the user three choices for each parameter makes claims more consistent. Also, it is more difficult to determine the difference between 0.6 and 0.7, but it is easier to distinguish between medium and high ratings. We combined several of the notification tasks and updated their definitions. The group thought that by combining similar choices the user would have an easier time making a decision. We implemented all of these changes in a new prototype.

The second round of testing was internally administered. To achieve accurate results, one group member was chosen to be the independent administrator of the experiment. This created an unbiased testing environment. For this round of testing, we chose claims that used more familiar artifacts, such as a cell phone vibrate feature and the sound notification in AOL Instant Messenger™.

Our internal testing validated the effectiveness of most of these changes. As the experiment progressed, it was evident that our group was more decisive due to the familiarity of the artifacts behind the claims. Additionally, a strong majority of the group was correctly able to identify the primary notification environment. The updated notification task titles and definitions proved to be the needed change to achieve consistency in this field.

The values achieved in testing the IRC values were more accurate than previous tests, but were not to the level that we desired. However, we were unsure of how to reengineer this aspect of the claim classification features since we did not have enough details about where accuracy and consistency broke down. Uncovering these details became the specific objective of the third round of testing.

5.3 Formal Lab Based Testing

To understand the breakdown details of the IRC classification scheme, as well as validate the consistency of other features for searching against stored claims, we decided to conduct a third round of testing.

5.3.1 Experimental Design

This experiment involved 11 students who were currently enrolled in the introductory HCI course so that all of them had recent experience with claims. The test was conducted in two phases: searcher and classifier. In the first phase, the participants (acting as *searchers*) were presented the claim-entering screen, a brief explanation of what a claim is, and the claim summary screen. The participants were given time to complete two claim searches during this portion of the test. This involved reading a problem and determining the low (0.0), medium (0.5), and high (1.0) IRC values, notification environment, notification tasks, and design concerns in order to retrieve a relevant claim from the library.

An example problem used in this phase is:

> *"Eric is designing an automobile computer system that will aid a driver in finding her destination using an already working GPS tracking component. Unfortunately, in the car Eric is building his system, he will not have any screen space available for a visual component. However, he wouldn't use one even if he could because he wants his system to be as safe as possible when used on the road. He wants the system to react as the user is driving, constantly updating the driver with information that will guide her to the destination."*

The second phase of the test involved the same participants (acting as *classifiers*) adding claims to the database. They were given existing claims and had to determine the IRC values using the web-based wizard. The distinction in IRC input methods (low, medium, and high vs. wizard) reflects the different levels of familiarity the two roles would be expected to have. A searcher would have a very general idea of appropriate constraints, while a classifier should be able to describe more subtle characteristics of the claim. The classifiers were also asked to identify the notification environment, notification tasks, and design concerns that would be entered into the library as part of the claim. The two phases were reversed for half of our participants so that we could obtain data on searching and classification results for all design problems.

5.3.2 Analysis and Interpretation

We focused our analysis on understanding the breakdowns of IRC specification, so we begin our discussion of this data. Since the testing also looked at the consistency of the other search indices, the latter portion of this section describes those results.

For the IRC parameter testing, recall that we had two groups of participants alternating as searchers and classifiers for claims on four problems. Also, the manner in which participants specified IRC values depended on the classifier/searcher role. To analyze classifier-to-searcher match tendencies, we calculated the differences between the decimal values obtained by the classifiers and the values the searchers submitted. The general process used for this comparison is depicted in Table 4, although the process was repeated for each of the 11 classifiers and searchers. The example data shown in Table 4 illustrates a probable claim hit (shaded, Classifier B-Searcher X), as well as how inferences were made on classifiers, searchers, and each of the parameters. Here, the Interruption (I) parameter is specified most consistently with an average difference of only 0.18.

Table 4. Process used to analyze classifier-to-searcher match tendencies showing example results for two classifiers and two searchers.

				Searcher X			Searcher Y			Classifier Overall
				I	R	C	I	R	C	
Classifier				0	1	1	0	1	1	
	I	R	C							
A	0.25	0.8	0.1	Avg Difference 0.28			Avg Difference 0.48			Unacceptable Avg Difference 0.38
	I	R	C							
B	0.1	0.9	0.6	Avg Difference 0.1			Avg Difference 0.27			Marginal Avg Difference 0.18
Searcher Overall				Marginal Avg Difference 0.19			Unacceptable Avg Difference 0.39			

The first result we found is that the overall average differences between searcher and classifier IRC values dropped 0.15 and 0.06 on second rounds for classifiers. This result suggests that, as classifiers became accustomed to the IRC system, their results became more consistent. We noted several examples where classification efforts would have resulted in probable claim finding by a searcher. Recalling that each of the eleven classifiers classified two claims, each of which were searched for by five or six searchers:

- Five classifiers would have had at least one claim found by a searcher
- Three classifiers would have had at least one claim found by two searchers
- One classifier would have had both claims found by a searcher

This shows that the system can be used successfully. However, overall classifier results showed an unacceptably wide range of IRC specification differences (overall std. dev = 0.27). In particular, the Interruption (I) parameter was significantly less consistently matched than the other two ($F(2, 357) = 4.48$, $p < 0.05$), suggesting the most critical need for reengineering.

These results may show differences in human performances and learning effect when classifying claims. When the same kind of experiment was run on expert users with a well-rounded knowledge of IRC parameters, however, results where much more consistent. There are two interpretations to this result. Either classifiers did not have enough understanding of the claim they were classifying or IRC classification requires a more expert understanding of the parameter specification process. To improve this, we suggest requiring a certain level of expertise before allowing specification of these values.

Also, searcher performances showed that given a specific problem, participants tended to look for the same IRC values when searching for a claim, especially if "medium" responses were disregarded. This implies that the interface for specifying search parameters should be limited to two value selections—"high" or "low"—with the addition of an option for "uncertain" specification (where that parameter would not be considered in the query).

The second portion of our analysis focuses on the concurrence between claim adding and retrieval, based on the following categories: notification environment, notification tasks and design concerns. Test results strongly favored successful search attempts in the system. Every search resulted in at least two hits in two categories when "OR" comparisons within a category were used to make matches, so this searching technique proved to be successful. Only 6 out of 66 classifications were not strongly matched by searchers (meaning that less than two-thirds matched the classifiers). The average hit rate per classification was 88.3% with a standard deviation of 19%. The notification environment and design choice categories both had very high match averages of 93%. Results for two of the design problems are shown; highest match rates are presented in Table 5a, whereas Table 5b shows the lowest.

Table 5. Percent of searchers agreeing with classifier specifications of claims

Classifier	Environment	Tasks	Design
1	100%	80	100
2	100	80	100
3	40	80	100
4	100	80	100
5	100	80	100
6	100	100	100

(a.) **Avg: 91% Std Dev: 16%**

Classifier	Environment	Tasks	Design
1	83%	0	100
2	83	66	100
3	100	83	100
4	83	100	100
5	83	100	100

(b.) **Avg: 85% Std Dev: 26%**

Overall, these results are very encouraging and point the way for the next step in iterative design.

6. CONCLUSIONS

Our claims library was developed to help people with the design process of notification systems. The biggest challenge we encountered was implementing an intuitive classification scheme that would help people find claims that correspond to their design concerns. Therefore, achieving consistency between claim adding and claim retrieval was an issue in designing the classification interface.

In the system we designed, there are two complementary types of classifications to find claims. The first one is finding a claim in terms of its notification environment, notification task, and design concerns. When using this scheme, our results were consistent and it is very likely that people will be able to find helpful claims. The second classification method we implemented was based on the IRC parameters of a claim. By specifying those parameters, our library finds claims that are closer to users' design models for their notification systems. Our results show specific breakdowns in consistency that can be addressed with interface design or user training. General results reported in section 5.3.2 showed that IRC values could be consistently specified and used for searching.

The focus of our future research will be on improving classification interfaces for claims. The classification based on notification environment, notification task, and design concerns should have additional categories or better sub-categories. The classification based on IRC values is the one that needs the most work. Redesigning our "Get Parameters" wizard and providing better examples for existing notification systems will be our first task. Broader reengineering for specifying IRC values must facilitate specification of the interruption parameter, the one achieving the least consistency. With this reengineering work completed, the interface will be ready to be put into full use for design work that is done through HCI classes and seminars. Studying facets of this actual use will be of interest to other researchers concerned with design knowledge reuse.

Acknowledgments

This work was part of a project for Virginia Tech Undergraduate Research in Computer Science (VTURCS). We would like to thank Sung Cho for his efforts.

References

[1] McCrickard, D. S. and Chewar, C. M. (2003). Attentive user interfaces: Attuning notification design to user goals and attention costs. *Communications of the ACM 46*, 3, ACM Press, pp 67-72.

[2] Sutcliffe, A. (2002). *The Domain Theory: Patterns for Knowledge and Software Reuse*, Erlbaum Assoc.

[3] Sutcliffe, A. G. and Carroll, J. M. (1999). Designing claims for reuse in interactive systems design. *International Journal of Human-Computer Studies*, 50, 3, Academic Press, pp. 213-241.

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

[5] Chewar, C. M. and McCrickard, D. Scott (2003) *Claims Reuse for Notification Systems Design: LINK-UP Vision and IRC Equations*. Technical Report 03-29, Computer Science, Virginia Tech.