THE UNIVERSITY OF CALGARY

Supporting Results Synthesis in Heuristic Evaluation

by

Donald Allan Cox

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

NOVEMBER, 1998

THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled “Supporting Results Synthesis in Heuristic Evaluation” submitted by Donald Allan Cox in partial fulfillment of the requirements for the degree of Master of Science.

Supervisor, Saul Greenberg, Department of Computer Science

Rob Kremer, Department of Computer Science

Jeff Caird, Department of Psychology

Andy Cockburn, Department of Computer Science, visitor (University of Canterbury)

Date

Abstract

This thesis defines results synthesis as a process and as an area of research within Heuristic Evaluation. Results synthesis is the process of creating a complete, coherent, and concise statement of the problems in an interface and their possible solutions from a collection of raw problem descriptions generated by a Heuristic Evaluation inspection. My research focuses on results synthesis as a participatory practice in environments supporting emergence. On the basis of observing groups performing results synthesis in paper based environments, I generated a “typical” scenario describing the process of results synthesis. Based on my review of the relevant literature and my observations, I put forth a set of requirements for any environment supporting results synthesis. These requirements are used as the basis for designing a real-time distributed groupware system to support results synthesis. The system is refined and found to be usable in an iterative formative evaluation.

Acknowledgments

My supervisor, Saul Greenberg, has been crucial to this research. He has provided invaluable assistance, direction, and motivation with both the big and the small issues. I could not, and certainly would not, have done this without him.

My thanks to the past and present members of the GroupLab for providing an amusing and stimulating work environment. In particular I want to thank Mark Roseman for all the GroupKit assistance, especially the 11th hour debugging help; Ted O'Grady for help in figuring out what a thesis was; Linda Tauscher for a sympathetic ear and reminders that there was life beyond the walls; and Carl Gutwin for letting me piggy-back on his research, demonstrating the pitfalls of quantitative evaluations and insightful last-minute feedback. Thanks for putting up with me. I hope I've done you proud.

My thanks to Jeff Caird, for valuing what I had to say, for forcing and encouraging me to expand my horizons, for thinking about process, for laughing at my cynical remarks, and for providing a counter-point to quick-and-dirty HCI research.

My thanks to those who participated in my numerous studies. Without them, my research is just so much navel gazing. It wasn't always easy, but with their help I think I've produced something of quality.

My thanks to Jas Chugh for providing a benchmark for measuring my graduate career, for making me glad I was not a psych. student, and for being a co-investigator and fellow sufferer.

I would also like to thank those who shared the "graduate student experience" with me: Dana, Rally, Kevin, Amy. Thanks for all the discussions and interest in my research, as well as making class bearable.

My thanks to Lindsey Roberts and Robert Rousselle for being a great pair of housemates. Thanks for all the inspiration and distraction.

My thanks to my parents for their belief in me, and their support and encouragement in all my endeavours.

My thanks to the Natural Science and Engineering Research Council of Canada and the University of Calgary for funding this research.

Dedication

To Mum and Dad. Thanks for all the love and support.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Introduction

In this thesis I will present my research on interface evaluation methodology. *Heuristic Evaluation* is a popular user interface inspection method (Nielsen, 1994a, 1995a), but the results synthesis stage (See Figure 1.1) is not currently well defined or supported. A haphazard approach to *results synthesis* leads to developers being presented with a report of problems in their interface that does not help them understand what the real problems are, and what they can, and should, do to address these problems (Sawyer, Flanders, and Wixon, 1996). Without a supporting environment, preparation of a quality report is very time consuming (Jeffries, 1994). My research defines results synthesis as a process and as a key stage in effective Heuristic Evaluation, as well as showing that environments can be constructed that support the process.

Heuristic Evaluation can be thought of as a single encompassing process or as combination of a number of subprocesses or *stages*. These stages include: preparations for the inspection; the inspection itself; results synthesis; and communication of the results to designers and developers (See Figure 1.1). Occasionally people will speak of “heuristic evaluation” when referring solely to the inspection stage. In this thesis, I will use “Heuristic Evaluation” to refer to the entire process.

Human-Computer Interaction (HCI) researchers have spent their effort in studying Heuristic Evaluation on the preparation (e.g. Nielsen, 1994a) and inspection stages (e.g. Nielsen, 1992), and how it compares with other evaluation techniques (e.g. Desurvire, 1994). This thesis concerns a different stage which I call results synthesis. Results synthesis is:

> The process of transforming the raw inspection data from Heuristic Evaluation into a complete, concise, and coherent statement of the problems in the evaluated interface as well as recommended actions to address the problems identified.

Results synthesis is a new area of research within Heuristic Evaluation. While other researchers have characterized the qualities of the output of results synthesis (Sawyer,

Heuristic Evaluation

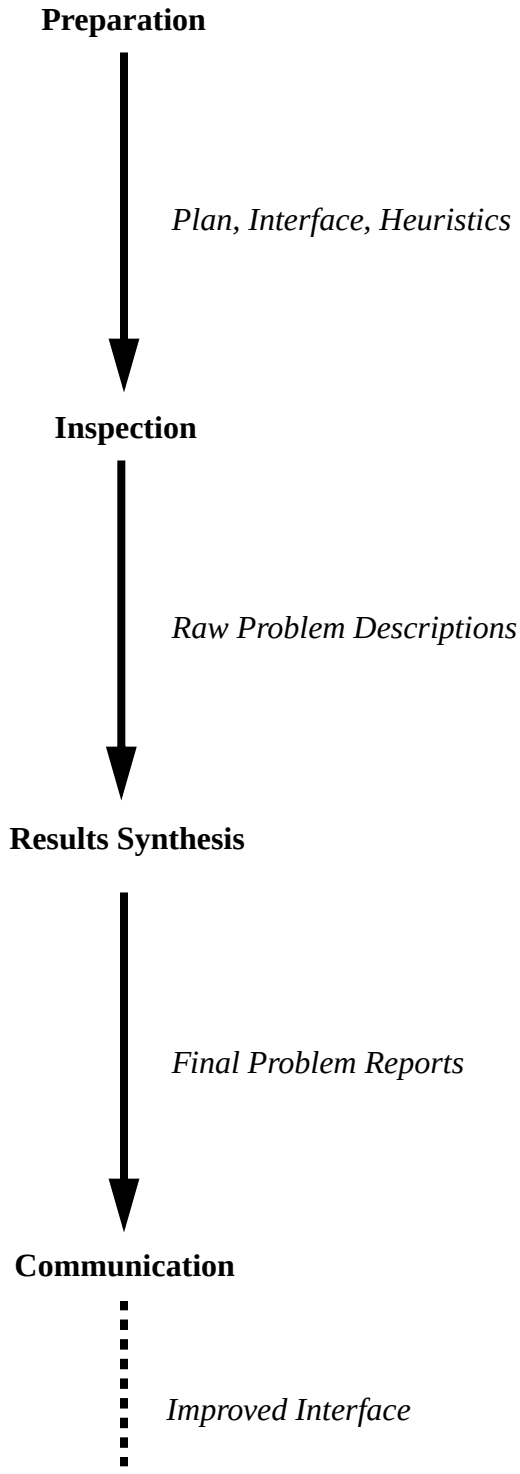


Figure 1.1: Heuristic Evaluation – process and *product*

Flanders, and Wixon, 1996; Jeffries, 1994), none have looked at the process of results synthesis, until now.

I chose the phrase “results synthesis” to communicate that what needs to occur in this stage is not a simple task of arrangement, selection, or categorization of problems identified from the inspection stage, but rather a process through which substantive new knowledge, understanding, and consensus is generated by the participants. If we start with the premise that the measure of a method’s benefit is its positive impact on the actual product (Sawyer, Flanders, and Wixon, 1996), it follows that results synthesis is an important area of research since it bridges the gap from the problematic raw inspection data (Nielsen, 1992; Jeffries, 1994) to a communication that is convincing to developers (Jeffries, 1994; Sawyer, Flanders and Wixon, 1996). What is new in my research is not the identification of the need to do this processing – practitioners have, of necessity, always been doing some form of results synthesis. My contribution is the identification of particular properties of results synthesis and the creation of environments specifically to support the results synthesis process.

In the next section I will discuss the three major themes upon which this research is based: Heuristic Evaluation, participatory practice, and emergence. Following that, I will provide my problem statement, research hypothesis, and research goals. Finally, I will provide an outline of the rest of the thesis.

1.2 Considering the research context

My research is centered on three concepts: Heuristic Evaluation, participatory practice, and emergence. In this section I will first place Heuristic Evaluation in the larger context of the software development enterprise and HCI/usability engineering in particular. Having established the general context for my research within Computer Science, I will present two of the major influences on my research. Section 1.2.2 presents a brief overview of participatory practice. The concept of emergence is covered in Section 1.2.3.

1.2.1 Heuristic Evaluation

Software development is often modeled as tri-partite cycles (see Figure 1.2) or spirals

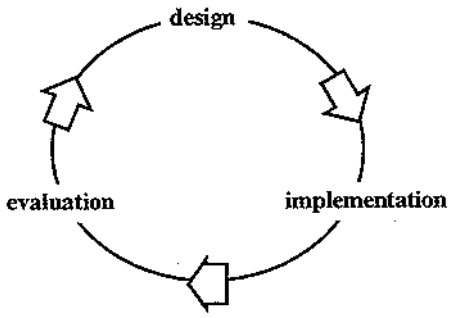


Figure 1.2: Tri-partite system development cycle (Greenberg, 1996b)

(Boehm, 1988). This conception of software development suggests that development activity consists of repeated sub-activities of design, implementation, and evaluation, where the output of the preceding stage is fed into the next stage. The cycle is arrested when resources run out or the product is completed. Any particular development effort can be subdivided along different lines into the "parts" of the software being developed – the user interface, the backend, the command interpreter, and so on. Each of these parts undergoes its own tripartite cycle. In the case of the user interface, the design sub-activity has its own tripartite cycle of design–prototype–evaluate.

In the evaluation part of that cycle there are a number of methods used to assess the design of a system's interface. Nielsen and Mack (1994) divide usability evaluation techniques into four general categories:

1. *Automatic* – usability problems are found by software analyzing some form of machine understandable specification of the interface.
2. *Empirical* – usability problems are found by testing the interface with "real users."
3. *Formal* – usability problems are found by calculation using "exact models and formulas."
4. *Informal* – usability problems are found by using "rules of thumb and the general skill, knowledge, and experience of the evaluators."

This last category, that of informal methods, includes *inspection methods* which Nielsen and Mack (1994) describe by saying that "In general, the defining characteristic of usability inspections is the reliance on judgement as a source of evaluative feedback on specific elements of a user interface." Depending on the inspection method, the inspectors use a set of rules, guidelines, or strategies to structure their search for problems in the interface.

One of the most popular inspection methods used in interface evaluation is Heuristic Evaluation (Nielsen, 1995a). This is due, in part, to its flexibility – there are many different ways of doing heuristic evaluations and practitioners are encouraged to tailor it to their particular situation. It is also promoted as a *discount usability* method (Nielsen, 1993). The idea of discount usability is that it is better to get some results using these methods when the alternative is doing nothing because the resources are not available to carry out more accurate but costlier methods. I have chosen Heuristic Evaluation as the basis for my research for two reasons:

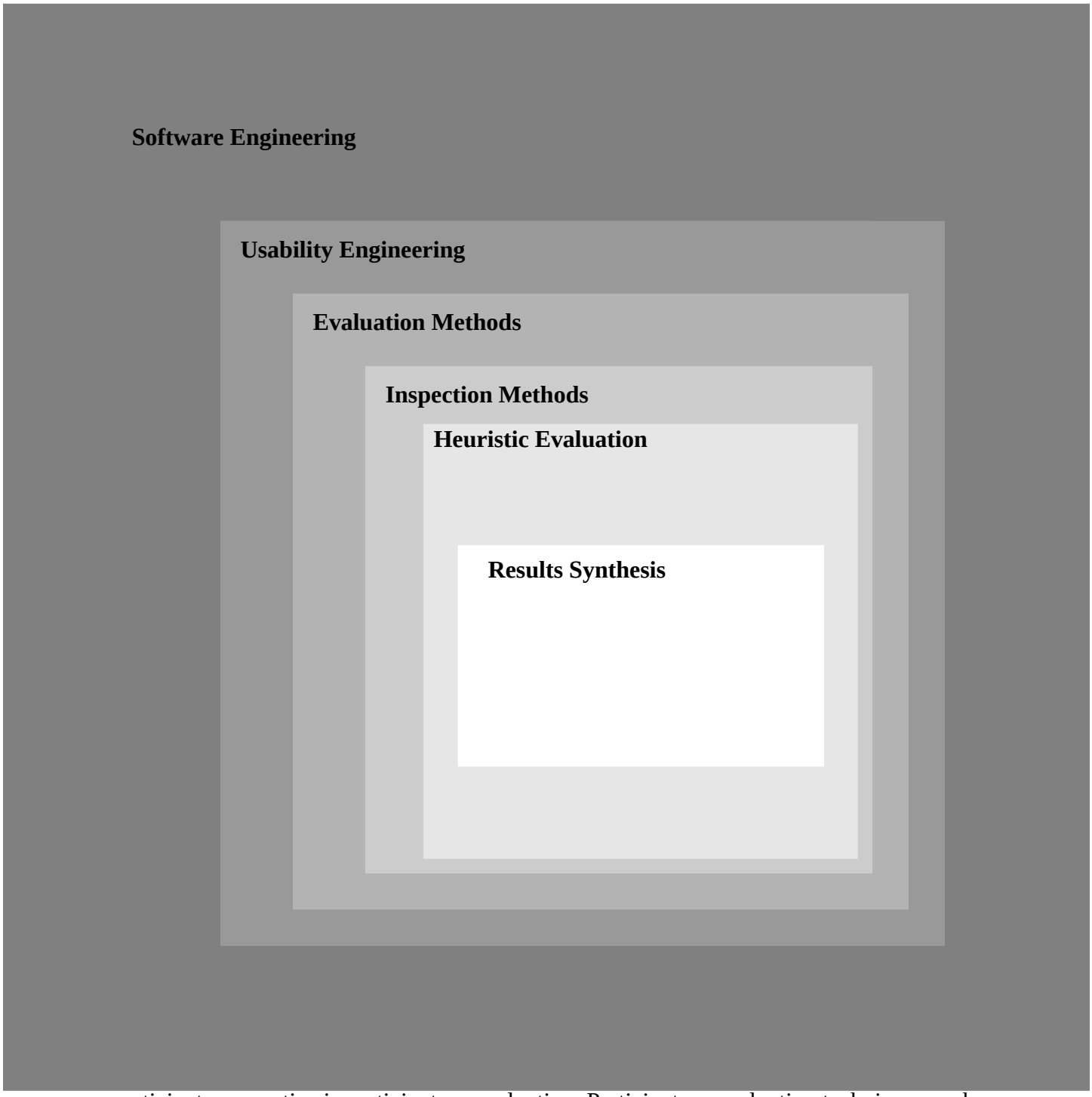
1. Heuristic Evaluation is popular in both academia and industry.
2. Results synthesis plays a key role in the quality of the outcome of a Heuristic Evaluation.

Figure 1.3 shows the relationship between Heuristic Evaluation and its enclosing disciplines. Heuristic Evaluation itself will be treated in more depth in Chapter 2.

1.2.2 Participatory practice

Participatory Design is a movement and a philosophy coming out of Scandinavia. Its premise is that the design of successful technology is more likely to occur if all stakeholders, particularly the eventual end users of the technology, are partners – participants – in the design of the technology. The users-as-participants are considered more than resources to be consulted or analyzed at the designers' whim. They are to be empowered and actively involved in the design decision-making process. The adherents to Participatory Design produced and published a number of techniques and methodologies for including users and others in design activities (e.g. Schuler and Namioka, 1993).

Researchers have subsequently recognized that other parts of user interface development can benefit from end user participation. They coined the phrase *participatory prac-*



participatory practice is participatory evaluation. Participatory evaluation techniques seek to place the user within the evaluation process as an expert in their own work, operating at the same level as the other experts taking part in the evaluation. In contrast, traditional user centered interface development processes involve end users in the evaluation stage only in a subordinate role as subjects in usability studies.

Recently, the idea of participatory evaluation was combined with Heuristic Evaluation to create Participatory Heuristic Evaluation (Muller, et al, 1998). This variant of Heuristic Evaluation diverges from the standard in two important ways:

1. Five additional heuristics are added to legitimize the social concerns of end users. This contrasts with the conventional heuristics that focus on people's efficiency in carrying out their work tasks.
2. End users become evaluators of the interface, contrary to Nielsen's expectation (Nielsen, 1994b). These end users are engaged as work domain experts, and as such provide additional insight into the problems of an interface.

Participatory practice has influenced my research by leading me to consider results synthesis as primarily a group activity. By doing so, results synthesis provides another opportunity for participation in a key step of the evaluation process. Results synthesis is a critical interpretive and analytic activity that creates the recommendations that affect the development of the interface being created. In my observations, questions about the users and their work often come up during results synthesis. Thus the presence of users in results synthesis would allow them to correct or confirm the picture of users developed during results synthesis. This closer correspondence to reality is expected to identify problems more quickly and reduce the number of surprises in later evaluations and the severity of the problems that are found.

1.2.3 Emergence

Emergence can be characterized as the observation that:

> Ideas do not arise well formed. At first there are expressions of fragments of thoughts. Once there is some rough material to work with, interpretations gradually begin to emerge as they are discussed. (Moran, Chiu, and van Melle, 1997, p. 46)

An example is the process Marshall and colleagues used in preparing a report on machine translation (Marshall and Rodgers, 1992). In this case, the participants spent some time before-hand coming up with what they thought was going to be the important types of information that they would need to gather and a scheme to organize it. However, in gathering the information and trying to make sense of it, they discovered that their precon-

ceived notions were not particularly useful for carrying out their actual task. During the process they had to come up with new ways of recording information and expressing understanding as they discovered new information, or new properties. As a result their understanding of what was important or meaningful evolved. This emergence phenomena has been observed in a number of different fields and there is increasing interest in how emergence might be better supported in systems (Edmonds, Moran, and Do, 1998).

Results synthesis is also characterized by emergence. My observations, and those of others (Monty, 1990; Marshall, Shipman, and Coombs, 1994; Shipman and Marshall, 1994), have lead me to believe that the understanding of the participants and the structure and nature of their belief about the problems in the interface emerges over the course of the process. Within results synthesis, the *raw problem descriptions* generated in the inspection are the "rough material" on which the participants base their work. One way to describe the problems in the interface is to present the collection of raw problem descriptions as defining the problems in the interface being evaluated. Another way would be to say that the problems in the interface are that the heuristics used in the inspection have not been heeded during design. The problem descriptions are then used as footnotes or explanations to the heuristics. Based on my own research and that of others (Jeffries, 1994; Sawyer, Flanders, and Wixon, 1996), I have concluded that neither of these ways of talking about the problems in the interface is satisfactory.

Results synthesis creates a description of the problems in the interface that is not present in either the collection of raw problem descriptions or in the heuristics used in the generation of the raw problem descriptions. This new way of understanding the problems in the interface is created by the participants as a result of discussing their "rough materials" and trying differing ways of organizing or categorizing the raw problem descriptions until they find one to their satisfaction. This final interpretation is one that emerges out of the interplay of materials, participants, and environment. Thus emergence, and the support for it, is key to results synthesis and my discussion of it.

1.3 Problem statement and research hypothesis

In practical application, Heuristic Evaluation is only as effective as its ability to influence developers, regardless of how many problems the inspectors discover in the interface. We know the qualities of problem reports that have a positive impact on developers (Sawyer, Flanders, and Wixon, 1996; Jeffries, 1994). Other researchers have not discussed how reports with these qualities are created from the raw problem descriptions produced by the inspection stage. Research on results synthesis will show how this transformation takes place and how it can be supported. Practitioners will also benefit from guidance about how to carry out results synthesis and how to support those carrying out the process in traditional or technological environments.

My research hypothesis is that results synthesis in Heuristic Evaluation is a definable and describable process, that constraints on the process may be identified, and that environments may be created that support the process.

1.4 Goals of the research

The goals of this research are:

1. Define and describe results synthesis.
2. Identify requirements for supporting results synthesis.
3. Construct and evaluate a prototype system for supporting results synthesis.

Each of these goals derives directly from a component of my research hypothesis. By providing a definition and description of results synthesis I show that it is definable and describable. In identifying requirements for supporting results synthesis I show the constraints upon the process. Constructing a system that supports results synthesis is an existence proof that environments supporting results synthesis are possible.

1.5 Thesis overview

Chapter 2 surveys the literature on Heuristic Evaluation. I present what is known about the various aspects of Heuristic Evaluation. This includes what heuristics to use, how many inspectors to use and what qualifications they should have, as well as the overall process of

performing an evaluation. Also, I relate what is known about how the results of Heuristic Evaluation are used to influence development.

A scenario describing results synthesis is presented in Chapter 3. This scenario shows a plausibly constructed instance of results synthesis in a paper-based environment based on my observations of real groups performing results synthesis. The chapter illustrates the entire results synthesis process from the preparation and collection of raw data to the preparation of final problem reports.

Chapter 3 gives the reader a general feel for results synthesis. In Chapter 4, I present the requirements for supporting results synthesis. These requirements are intended to ensure that the raw data is in an appropriate form for use in the process, that results synthesis produces quality output, that results synthesis occurs in an emergence-enabled environment, and that all the necessary information is recorded at the end of the process. The requirements also cover the different types of participants and how they are to be included.

These requirements are used to drive the design of a groupware system supporting results synthesis. In Chapter 5 I present and illustrate the main design decisions faced in creating the system, tracing their origin to particular requirements. These decisions center around how the content will be represented in the workspace, how the workspace is presented, and how the users will interact with the content in the workspace.

Chapter 6 chronicles the evolution of the interface through formative usability studies where the results of individual studies are used to progressively refine the interface. I conducted eight studies, three in a single user scenario, four in a two person group scenario, and one is a three person group scenario. For each study, I present the major findings, the changes made to the interface, and the unresolved questions raised.

Finally, I summarize the results of my research in Chapter 7. The major results are the definition and description of results synthesis, the promulgation of requirements, and the creation and evaluation of a system to support results synthesis. Areas for further research are also identified.

Chapter 2: Heuristic Evaluation

2.1 Introduction

This chapter presents an overview of the goals and method of Heuristic Evaluation and a more detailed consideration of how it might be made more effective.

Software developers are constantly being asked to produce more with less. Although software engineers may know the “right” way to build a software product (Humphreys, 1989, 1995), they seldom have the resources — staff, budget, time, expertise — to carry out the ideal plan. As part of the engineering lifecycle, the interface design, implementation, and evaluation activities carried out by usability engineers are faced with the same pressures, perhaps to an even greater degree. This is because usability engineering is often considered more marginal than other software engineering activities (Bias and Mayhew, 1994). One response to this pressure has been the creation of discount usability methods or techniques (Nielsen, 1993). The notion behind discount usability is that there are cost-effective ways of performing usability evaluations in even the most resource starved projects. While these methods may not provide optimal detection of usability problems, they provide a good return for the effort invested.

Heuristic evaluation is the prototypical discount usability method. It uses a small number of relatively easily procured inspectors to inspect an interface at any stage of its specification, design, or implementation. Using a small set of heuristics, these inspectors look through the interface and describe the problems they see with reference to the heuristics. These problems are then communicated to those developing the interface.

In this chapter I consider Heuristic Evaluation as it is presented by its main proponent, Jakob Nielsen. In Section 2.2, I describe the recommended best practices for Heuristic Evaluation and how they came about. Where relevant, I include opinions, studies and experiences of other researchers and practitioners of Heuristic Evaluation. I then look at the practicalities of connecting Heuristic Evaluation to the rest of the engineering lifecycle in Section 2.3. In Section 2.4 I return to a more detailed consideration of the part of the Heuristic Evaluation process I call results synthesis.

2.2 Heuristic Evaluation

Heuristic Evaluation can be characterized as

> having as small set of evaluators examine the interface and judge its compliance with recognized usability principles (the “heuristics”) (Nielsen, 1994b, p. 26).

In this section, I will examine the main elements of Heuristic Evaluation as contained in the above statement: the interface, the heuristics, the inspectors (evaluators), and the judgment of compliance. Let me first place Heuristic Evaluation within the context of usability evaluation methods.

When describing usability evaluation methods, there are two major themes. The first theme is user testing (Dumas & Redish, 1993; Rubin, 1994), wherein problems are found by observing a sample of users doing work with the interface under evaluation. The other major theme is inspection. Usability inspection methods find problems by having a small number of experts examine the interface (Nielsen & Mack, 1994). There are also two loosely defined categories of usability inspection methods. One is walkthroughs, where the inspectors use task examples to methodically evaluate the interface based on a theoretical model. Examples from this category include Pluralistic walkthroughs (Bias, 1994), and Cognitive walkthroughs (Wharton, 1994). The second category contains methods where the inspectors base their problem detection on their experience and a set of principles or guidelines and inspect the interface in a more or less structured manner. Heuristic evaluation (Nielsen, 1994b) and Formal usability inspections (Kahn & Prail, 1994) are examples of methods in this category.

The rest of this section will present a definitive account of Heuristic Evaluation based on the published record. It will examine, in order, the role of the interface, the heuristics, the inspectors, and the process of examining the interface. These are presented in roughly the same order in which they appear when inspectors apply Heuristic Evaluation.

2.2.1 The interface

Inspectors performing Heuristic Evaluation require an interface to evaluate. Nielsen & Mack (1994) suggest that Heuristic Evaluation in particular, and usability inspection

methods in general, are especially suited to evaluating an interface early in the design cycle, before implementation activities have produced a working version of the interface.

Of course, usability evaluations may occur at any point in the engineering lifecycle. When they are performed on a completed interface, they are referred to as summative evaluations. The purpose of a summative evaluation is to measure the interface against the goals set at the beginning of development. In contrast, formative evaluation is performed on an incomplete or evolving interface with the intent of using the results of the evaluation to inform changes to the interface. Heuristic Evaluation is generally used in situations where a formative evaluation is sought. In particular, it lends itself to use in the early stage of design. Interface construction is an iterative process where the description of the interface gradually becomes more concrete and more operational. As the interface moves through the lifecycle, it is represented by a series of prototypes of increasing “fidelity” (Rudd, Stern, & Isensee, 1996). Low fidelity prototypes are hand drawn representations of the interface that are quickly sketched out to give a feel for the flow of the interface and have no programmatic behaviour, but rely on the designer to simulate the intended behaviour in the interface, for example by moving bits of paper around. High fidelity prototypes are running programs that implement much of the appearance and the important behaviours of the finished product. Between high and low there is a broad range of what might be called medium fidelity prototypes. The PICTIVE technique (Muller, 1993) is at the boundary between low and medium fidelity prototyping as it uses both pre-formed elements as well as hand drawn ones. A Powerpoint™ slide show might be on the boundary between medium and high fidelity prototyping. Heuristic evaluation is often portrayed as best for low and medium fidelity prototypes with the thought that the expert inspectors are able to concentrate on high level interaction and organization issues.

A caveat to the use of Heuristic Evaluation to evaluate interfaces early in the design process is that the inspectors tend to miss certain classes of problems, such as those arising from perceptual-motor slips (Mack & Montaniz, 1994), or missing functionality (Nielsen, 1992). This may be compensated for to a degree by directing the inspectors to pay particular attention to these areas. Other evaluation techniques, such as usability testing, can be

applied to early interface designs (Karat, 1994), perhaps uncovering other problems and therefore complementing Heuristic Evaluation. While Heuristic Evaluation is not perfect, it does find many problems, including important ones, making it a very useful and widely used method.

2.2.2 Heuristics

Heuristics are usability principles that inform the Heuristic Evaluation process. The challenge is to produce an effective set of heuristics. Nielsen and Molich (1990) recommend that a good set of heuristics will be small (e.g., around 10), so that the inspectors will have an easy time remembering, being reminded of or referencing them, while still being rich enough to detect a large number of usability problems. Figure 2.1 lists the ten heuristics currently being recommended by Nielsen along with their secondary text. These heuristics have evolved from those originally proposed (Nielsen and Molich, 1990, Molich and Nielsen, 1990). The original set was chosen based on the Nielsen's and Molich's understanding of typical problem areas of usability, as well as an informal consideration of existing guidelines. The list of heuristics in Figure 2.1 is a result of a more formal factor analysis. In this analysis, Nielsen (1994a) studied how well a large number of principles, chosen from the original set of heuristics as well as six other collections of published principles or guidelines, accounted for usability problems found in the database of problems collected in the course of his earlier studies of Heuristic Evaluation. Seven of the ten new heuristics explain a significant percentage of the problems found in the database. The other three are included because Nielsen feels they are important, based on his experience.

There are a number of unanswered questions about heuristics. Nielsen (1993) mentioned that additional heuristics may be added, but that they are to be domain specific ones. Muller et al (1996) recommended the addition of four more "participatory" heuristics to Nielsen's current ten. They present an empirical justification for three of these and theoretical basis for all four. The list has been further refined and reorganized (Muller et al, 1998). There has never been a solid theoretical basis for the heuristics used, and Muller et al's (1995) findings of the utility of more heuristics cast doubt on the notion that Nielsen's list is as good as we could get. Additionally, the exact role or roles played by the heuristics

Ten Usability Heuristics

Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Match between system and the real world

The system should speak the users' language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

.

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

.

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

Recognition rather than recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use

Accelerators — unseen by the novice user — may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experience users. Allow users to tailor frequent actions.

Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Figure 2.1: Nielsen's ten recommended heuristics from (Nielsen, 1994b, p. 30)

in the evaluation process has not been spelled out. While it would be reassuring to have good answers to all the questions about Heuristic Evaluation, we know enough to use, recommend, and teach the method.

2.2.3 Inspectors

The productivity of Heuristic Evaluation depends upon the number and the type of inspectors taking part in the process. In their initial presentation, Nielsen and Molich (1990) showed that more inspectors used means more problems found. This has been confirmed in subsequent studies (Nielsen, 1992). From the discount usability perspective, an optimal trade-off is sought, as there are diminishing returns as more inspectors are used (see Figure 2.2). With fewer than three inspectors, many usability problems are not identified.

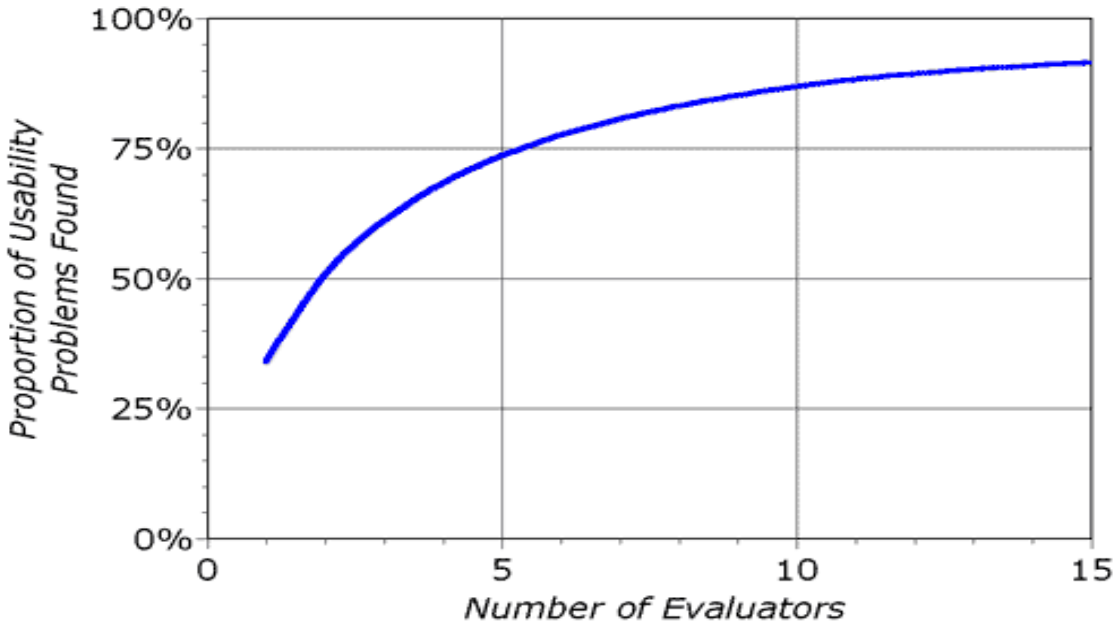


Figure 2.2: Curve showing percentage of problems found versus number of inspectors used. From (Nielsen, 1994b, p. 33)

Increasing the number of inspectors beyond five does not yield an increase in problems detected proportional to the added effort. Consequently, Nielsen recommends three to five inspectors as providing the best cost/benefit trade-off.

There are two dimensions of inspector expertise that are relevant to Heuristic Evaluation. The first dimension is expertise in generic interface usability. Nielsen (1992) refers to

these experts as “usability specialists.” The second dimension of expertise is experience with the special issues to be found in the particular type of interface being developed. For example, this could be previous experience in the design and evaluation of telephone voice response interfaces (Nielsen, 1992). Nielsen’s study (1992) found that “double expert” inspectors – those with expertise in both general usability and telephone response systems in this instance – performed better than those who are only usability specialists, who in turn perform better than minimally trained inspectors.

However, even usability specialists are a scare resource (Sawyer, Flanders, & Wixon, 1996). Nielsen (1995a) reports that even when usability activities are well funded, few practitioners use more than three inspectors, and many use only one or two. One approach to getting more inspectors is to use less expert inspectors (Gunn, 1995). This reality has lead to an interest in how the less experienced may still be fruitfully included in the evaluation. Desuivre (1994) proposed a role-playing technique to improve the problem-finding ability of inspectors. Nielsen (1995b) suggested the use of scenarios to make up for lack of domain knowledge. Muller et al (1998) recommended including inspectors other than usability experts on the basis that they bring perspectives or knowledge not otherwise available. In essence, while performing Heuristic Evaluation using inspectors who are not double experts might be non-optimal, it is still better than doing nothing at all (Nielsen, 1993).

2.2.4 Exercising judgment

Nielsen (1994b) recommends that inspectors independently inspect the interface by following a particular process. The recommended best practice is to have the inspectors make two inspection passes through the interface. The purpose of the first pass is to gain an overall familiarity with the interface. The second pass consists of a detailed examination of all aspects of the interface with respect to the list of heuristics. As previously mentioned, problems in the interface are located by the inspectors using their experience and the heuristics as a guide. To some, this makes the result of Heuristic Evaluation mere opinion, and hence of less value than empirical or theoretically derived results (Sawyer, Flanders, & Wixon, 1996).

The problems should be recorded indicating what the problem is, where it is, and why it is a problem, including citing the applicable heuristic. Each inspector considers the interface independently, without any interaction with the other inspectors. The goal behind this is to allow the widest possible discovery of problems when the individual results are aggregated (Nielsen & Molich, 1990). The reason for having the inspectors work separately is the belief that if they were allowed to interact, this would result in a channelization of their attention, and as a result, fewer problems would be found.

The Heuristic Evaluation process has evolved over time. Originally, the inspection process was described as just looking at the interface and forming opinions (Nielsen & Molich, 1990). This level of guidance is likely adequate when usability specialists are evaluating the simple interfaces offered by the examples in the early studies of Heuristic Evaluation (Molich & Nielsen, 1990, Nielsen & Molich, 1990). As Heuristic Evaluation has matured, various researchers offered more detailed descriptions of how the inspectors are to carry out their mission. Karat (1994) used a two phase inspection process. In the first phase, the inspectors engaged in “self guided” exploration while in the second they used scenarios for programmed exploration. Desuirve’s PAVE (1994) advocates several inspection passes through the interface. In each pass the inspector considers the interface from a different point of view taken from a defined set of roles.

As already mentioned, Nielsen is insistent that the inspectors carry out their inspections separately and independently so as not to bias each other. However, Sawyer, Flanders, and Wixon (1996) recommended using pairs of inspectors, but do not provide any empirical evidence to support this position. Karat (1994) also recommended the use of paired inspectors as more effective and efficient based on the results of her study of a Heuristic Evaluation variant. It is difficult to know which approach is best as no single, unified metric or set of experimental conditions has been used to compare the various approaches, variations, and ideas (Muller, Dayton, & Root, 1993). Thus, the different results and recommendations cannot be integrated or reconciled. This suggests that the biases of the practitioner, considering his or her situation, will determine the particular variant of Heuristic Evaluation used.

2.3 Connecting to development

Once the inspectors have completed their separate inspections, the lists of problems are aggregated into a unified list of all the problems. This list is given back to three or four of the inspectors, who judge the severity of each problem on a five point scale. These severities are then averaged, and the resulting annotated problem list is communicated to the developers of the interface for them to fix (Nielsen 1994b).

Aggregation is the process by which the content to be communicated is refined from the raw, unusable data produced by the inspectors. This process of aggregation is central to the effectiveness of Heuristic Evaluation (Nielsen & Molich, 1990). The effectiveness of Heuristic Evaluation is based on the ability of its practitioners to influence product development. This influence is exerted through communication with those who actually effect the development.

The use of the term aggregation does not adequately reflect the complexity of what actually goes on in the process of turning the disparate descriptions of problems produced by the individual inspectors into a complete, coherent, and concise whole that will have the desired impact on the development of the interface. I therefore use the phrase results synthesis to refer to this process. There has been little study of this process or guidance given as to how to perform this most important of tasks. Nielsen (1994b) limits his suggestions to saying that it may be carried out either by an individual or by a small group. In one published report the results synthesis process reduced the number of problems by about two-thirds (Muller et al, 1995), which is in keeping with my own personal experiences and observations. Given that this process is responsible for turning the raw evaluation data into something that will have a positive impact of the development of the interface, it is important research area. This is the focus of this thesis.

2.4 Researching results synthesis

I define results synthesis as:

> the process of transforming the entire collection of raw problem descriptions into a complete, coherent, and concise statement of the problems in the evaluated interface as well as recommended actions to address the problems identified.

This process is more than simply concatenating lists of problems reported by the individual inspectors and the removal of obvious duplicates. Rather, it is the transformation of a number of opinions into a set of complete, coherent, and concise recommendations about how the interface can be improved. This represents the generation of new knowledge about the interface as well as its context of use and production. Within this definition one may imagine many different variations on the process that may be enacted. In this research, I chose to focus on one particular approach to results synthesis, one that is participatory in nature. In the rest of the section I explain why research on results synthesis is important and why a participatory approach is a good idea.

I chose to focus on results synthesis in this research as it has received little attention and I believe there are benefits to be gained from a more considered approach to it. There are several ways of leveraging the results synthesis process so that it provides the maximum benefit to the organization and the product. The first is to make the result of Heuristic Evaluation be a list of solutions (Nielsen, 1994b) or recommendations (Sawyer, Flanders, & Wixon, 1996) instead of problems. This approach is also used successfully by Muller et al (1995) in affecting the development of the interface they evaluated.

The second way of leveraging the results synthesis process is to involve directly the developers, documentation specialists, and customer support engineers (Nielsen, 1994b). Their participation ideally leads to an effective, well supported set of recommendations. In a non-participatory approach to results synthesis, these stakeholder groups would be consulted after the fact to get their input and feedback, which may be incorporated into the final report. This is often a time-consuming process and has the potential for generating confrontation amongst various groups if they feel they are being unfairly criticized or ignored. In a participatory approach to results synthesis the stakeholder concerns are addressed as a part of process of creating the final report. This reduces or eliminates the need to get feedback from these groups after the recommendations have been put forward, as their points of view have already been incorporated into the report. Given that usability concerns are only one of the demands for development resources (Sawyer, Flanders, &

Wixon, 1996), the combination of these two improvements provides product benefit by helping to ensure that the best trade-offs are made amongst those competing demands.

A participatory results synthesis process also provides an added organizational benefit by educating those involved who are not usability specialists about usability concerns and educating those who are about the concerns of the other stakeholders. The results synthesis process provides an important opportunity for learning. Inasmuch as the process involves discussion about usability, both in general and in the specific case of the interface being evaluated, it provides an opportunity for non-usability specialists to learn about usability, which will make both the products they are subsequently involved in developing better at the first cut, and the participants more effective evaluators in the future Heuristic Evaluations. The participatory approach also helps the usability specialists to refine their knowledge of the end users and the constraints faced by developers.

An additional organizational benefit of the expanded results synthesis process is that usability activities are seen as realistic, positive contributions to the development effort. One way this is manifested is through the elimination of false alarms (Jeffries, 1994). False alarms are reports of problems that are not in fact problems with the interface, perhaps because the inspector is mistaken or unaware of the constraints on the interface. An example of this is the recommendation to change an interface that would lead to inconsistency with other applications in a suite (Sawyer, Flanders, & Wixon, 1996). This winnowing of the proposed problems is important before the list of recommendations is formally made available to the developers to maintain the credibility of usability engineering.

At this point the reader may be suspicious of the goal of this research – suggested improvements to Heuristic Evaluation – when practitioners are unable or unwilling to meet the minimum recommendations of the existing process, particularly in terms of the number of inspectors used (Nielsen, 1995a). According to Nielsen's survey (1995a), the respondents were part of well funded usability efforts, so the problem must not be one of strictly dollar cost, but of other resource or cultural issues. An example could be the difficulty of getting a group of qualified evaluators who can meet face-to-face. If this is the case, then one way overcoming this obstacle is through the introduction of results synthe-

sis support in the form of a groupware system. While groupware technology is not a panacea (Grudin, 1988), it may reduce the perceived and actual costs of assembling a group for the purposes of performing results synthesis.

Some knowledge about results synthesis can be gleaned from the literature even though it has not be formally studied. What goes on in result synthesis most likely depends to some degree on what the final output of the evaluation is to look like. In this regard Jeffries (1994) has described the desirable properties of individual problem reports (See Figure 4.2 on page 56 for an example of Jeffries' report style). Amongst the characteristics she enumerates are:

- separation of problem and solution;
- textual description of severity;
- justification for problem and solution;
- provision of alternate solutions;

Muller (1997) has suggested that the problems are best if directly entered into the problem management system used to track all other problems with the software which has been traditionally reserved for those found in traditional software quality assurance testing and the problems reported by customers. Much more can be learned about results synthesis and the knowledge gained will lead to a more effective Heuristic Evaluation process, both theoretically and in practice.

2.5 Conclusion

Heuristic Evaluation is a widely accepted method for diagnosing usability problems in user interfaces (Nielsen, 1995a). Its level of acceptance with the research community can be seen by its inclusion in many HCI textbooks (e.g. Preece et al, 1994) and HCI courses (e.g. Greenberg, 1996a). Heuristic Evaluation is a usability inspection method wherein three to five expert inspectors make independent assessments of problems in an interface. These assessments are then aggregated to produce a more accurate picture of what is wrong with the interface. While there is general agreement about what Heuristic Evalua-

tion is at a high level and how it ought to be conducted, there are many open questions about the most effective way to perform Heuristic Evaluation, and how to compensate for the compromises that arise in practice. Particular areas of investigation and discussion amongst researchers and practitioners are the properties of inspectors, the heuristics, the details of the process of inspection and the best form for the final result. The goal of my research is to look at how to best support the results synthesis stage of Heuristic Evaluation, particularly when it is being carried out by a participatory group.

Chapter 3: Illustrating Results Synthesis Activity

3.1 Introduction

In this chapter I present a scenario to illustrate the results synthesis process and how it is carried out in a particular paper-based environment. My goal is to give the reader a sense of the general shape of the process and the expected behaviours that constitute it. I will not explicitly speak of requirements in this chapter. However, the reader should be able to discern connections between the requirements that will be listed in Chapter 4 and what goes on in this scenario.

This scenario shows how results synthesis may be carried out using paper as the primary organizing and recording media. My rationale in using a paper based environment is that the participants will find it a natural, open, and inviting place in which to do the work. Using paper is a way to encourage participation, especially when some of the participants are not technologists. It allows people to draw on their general experience and natural inclinations in dealing both with the materials as well as the other participants.

The scenario is divided into four stages, each corresponding to a section in this chapter:

1. **Preparation**: The participants gather the necessary raw data, and prepare a suitable space for results synthesis (Section 3.2).
2. **Familiarizing**: The raw data is arranged on the work surface, and all participants review the entire collection (Section 3.3).
3. **Emergence**: The participants iteratively reorganize and refine the organization of the data on the work surface to model what they believe is the best way to conceptualize the problems of the interface being evaluated and how to solve them (Section 3.4).
4. **Finalizing**: The participants record a rich interpretation of the layout into a standard report format (Section 3.5).

The scenario is intended to give the reader a sense of what it is like to participate in results synthesis. It is not a strict definition of this process. Neither is it a transcript of an actual session. Rather, it is a plausible construction based on my observations of real groups performing results synthesis in a paper-based environment.

After presenting the scenario, in Section 3.6, I discuss my observational studies of groups performing results synthesis in paper based environments and how they differed from the scenario I present in this chapter.

3.2 Preparation

The preparation stage begins when the decision to use heuristic evaluation is made, and ends when the participants actually meet to complete the results synthesis activity. The preparation stage consists of two activities. The first activity is ensuring that the raw data – the problems the inspectors find in the heuristic evaluation – is recorded in a suitable fashion. I assume a certain model of Heuristic Evaluation where the inspectors are responsible for recording the problems they discover (Nielsen, 1997) and that they will be participants in results synthesis. In the rest of this chapter, when I refer to inspectors and participants I am referring to different roles, though I expect them to be the same people. The second activity is the creation of a work space that will support the participants in carrying out results synthesis. These two activities will be discussed in turn.

3.2.1 Ensuring the data is suitably recorded

When performing results synthesis using paper, we first require that inspectors record the problems they find in the inspection stage in a manner that will later allow results synthesis to proceed smoothly through all its stages. These requirements are in accord with what is done in traditional Heuristic Evaluation inspection processes and do not represent an additional onerous burden being placed on the inspectors.

- Problem descriptions are written one per piece of paper.
- The pieces of paper are of a size and stiffness such that they can be easily stuck to the work surface, easily grasped when on the work surface either singly or in a small cluster, and subsequently repositioned without tearing or excessive delicacy.
- The pieces of paper are big enough to record all the desired information.
- For each problem found, the inspectors record a description of the problem as well as the heuristic they associate with the problem. They may also note any ideas about

potential solutions to the problem. This is the irreducible unit of raw data for results synthesis – the raw problem description.

- The inspectors record each problem description in such a fashion that it can be read from a distance of approximately one meter. Having the inspectors hand write the problem descriptions in large text and using felt tip markers has proved adequate in practice.
- The participants are able to identify which problem descriptions came from which inspector. For example, inspectors can each use a different colour ink to write their problem descriptions. However, differences in handwriting are usually sufficient.

In this scenario, preparation begins by distributing the materials to be used in the inspection to the inspectors (Figure 3.1). These include a description of the interface, materials

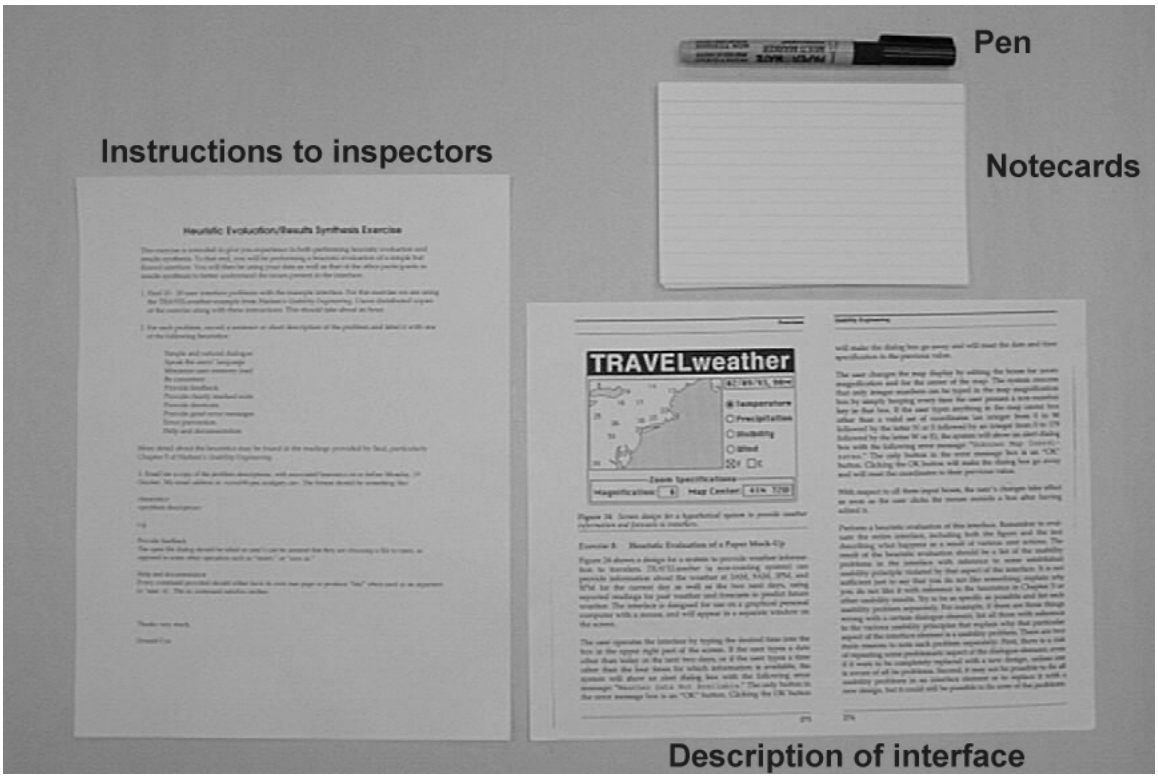


Figure 3.1: Materials given to inspectors

for recording the problem descriptions, and directions on how to perform the inspection and what heuristics to use. The interface being evaluated is the one given in Nielsen's (1993) *Usability Engineering* as Exercise 8 (p. 273-4) (See Appendix A for a reproduction

of the description). This particular interface has a single screen. While apparently simple, Nielsen has identified thirty-one usability problems in the interface. A snapshot of the screen plus a description of the interface's behaviour is contained on the sheet in the bottom right corner. In the upper right of the figure is a stack of 4x6 index cards and a medium-tipped marker pen. The index cards are used to record the problems found during the inspection; in practice they have been the right size for both recording problem descriptions and for later use in paper-based results synthesis. The medium-tip marker pen encourages the inspectors to write problem descriptions that are easy to read at a distance. The sheet at the left of Figure 3.1 contains instructions that tell the inspectors all they need to know to prepare for results synthesis – what heuristics to use, how to record the problems found, and other details of the evaluation (Appendix B).

At this point, the inspectors take the materials and perform the inspection of the interface individually according to standard Heuristic Evaluation methodology. Enough time has been allowed for the inspectors to do the inspection, taking into consideration their other workload.

Figure 3.2 shows three of the problem descriptions completed in this scenario by three different inspectors, illustrating the variation that occurs is practice in how the requirements are met. The problem description at the top has the heuristic written out fully at the top of the card. The description of the problem is fairly brief. The card includes a note at the bottom of the card indicating a possible solution. The middle card also has the heuristics – in this case there are two of them – at its top, but in a much abbreviated form. The description of the problem is in point form, where it notes a couple of closely related issues. The bottom card has the heuristics noted at its bottom, and again two of them recorded. The description of the problem is more verbose than in the other two. All of these are acceptable inputs to results synthesis, as the participatory process is able to make full use of these varying inputs.

3.2.2 Configuring the workspace

When the participants in results synthesis meet, it signals the end of the preparation stage and the commencement of the next stage. In order to have this meeting, they have to have

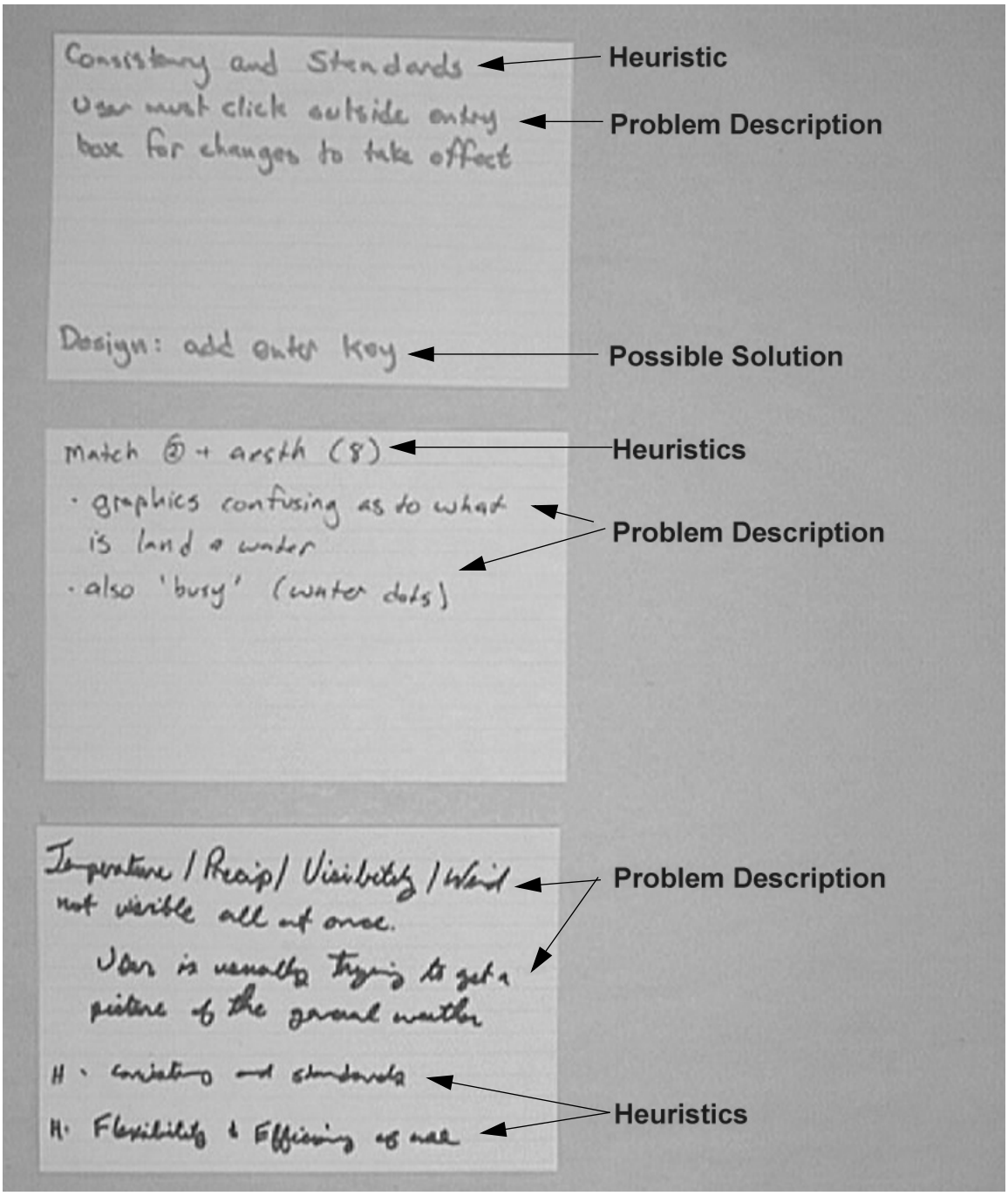


Figure 3.2: Three examples of completed problem descriptions

a place to meet. The preparation of this meeting place is the second activity of the preparation stage, and can happen independently of the inspection activity. Typically, however, there a gap in time between when inspection is completed and the participants meet. It is

in this gap that the workspace is set up. A suitable space for performing results synthesis using paper is a room with a door that closes, plenty of free wall space, and a meeting table.

The room should have at least eight feet of free wall space. When using paper, results synthesis uses wall space as the workspace. I have taken to covering wall space with large sheets of paper, upon which the participants will affix the problem descriptions and make annotations in the course of the process. Putting the paper on the wall makes it easier to remove and to store the end result, as well as to create new sections of workspace if the participants find they need more space than they anticipated, or annotations on the work surface become meaningless or obsolete. The paper also provides some protection to the underlying surface. When 4x6 index cards are used for recording problem descriptions, an eight foot by four foot section of wall space proves adequate for dealing with about sixty problem descriptions. For every additional twenty problem descriptions, you will need to add another two foot by four foot section to the workspace.

The room should also have a small table and enough chairs to seat all the participants. The table and chairs must leave enough space for the participants to move freely about in front of the workspace without getting in each others way. They should be able to stand back and see the whole workspace. There should be no impediments to moving from one end of the workspace to the other. The table provides a spot for gathering at the beginning of the process as well as a place to put materials that are not being used on the workspace at the moment. If a facilitator or recorder is present, they can sit at the table so as to be out of the way, yet still involved. Sitting at the table provides an overview, and avoids the temptation for the facilitator or other observers to get over-involved in the particulars of the process. Using a wall as the work surface instead of a table eliminates orientation problems – if people stand around a table, the content is going to be right-side-up for only a fraction of the participants – and provides more usable area for the work surface.

In this scenario, once the inspectors have created all their problem descriptions they meet in a room to perform results synthesis. Figure 3.3 shows the room used in this scenario as it appeared before the participants arrived. In the foreground is a table with some

Figure 3.3: A room prepared for results synthesis

additional materials on it. In the background is an eight foot long whiteboard that has been covered with large sheets of newsprint to form the work surface. Figure 3.4 shows the materials on the table in close-up. Of particular note are the two tubes of temporary adhesive standing up-right on the table. They will be applied to the backs of the index cards so that the cards can be stuck to the work surface and repositioned as a part of the results synthesis process. There are also marker pens and sticky notes for annotating the work surface, as well as additional index cards for annotating or adding additional problem descriptions to the work surface.

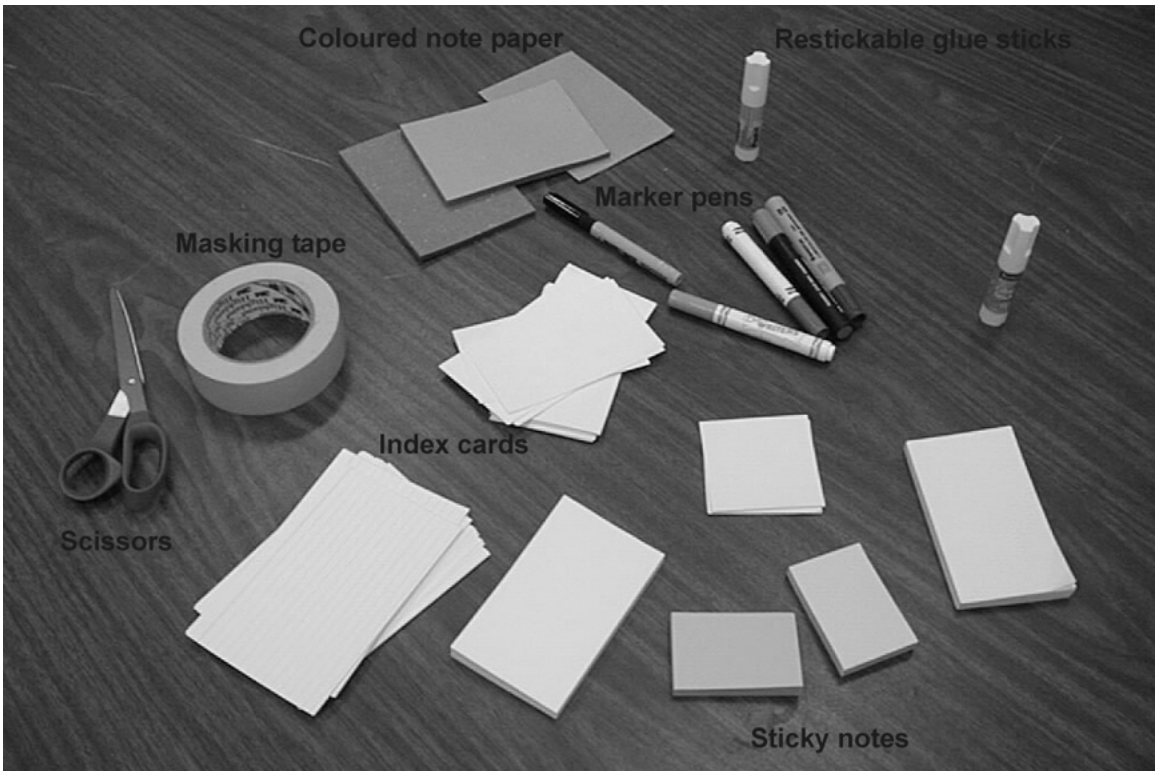


Figure 3.4: Materials provided to participants in results synthesis for use in the process

3.3 Familiarizing

Familiarizing has two goals: to prime the workspace with the entire collection of problem reports created during the inspection; and to have the participants in results synthesis scan every problem report.

3.3.3 Priming the workspace

In this paper-based scenario, the participants arrive bringing the problem descriptions they had each recorded individually. The first order of business is to prime the workspace. This is done by the participants sticking all fifty-eight problem descriptions they had brought onto the large sheets of paper that comprise the workspace. Note that the problem descriptions used in this scenario are a subset of those produced by the participants in my first observational study, which will be discussed further in Section 3.6. Appendix C reproduces all of the problem descriptions used in this scenario. Figure 3.5 shows the workspace with all the problem descriptions placed in it.The participants group the cards

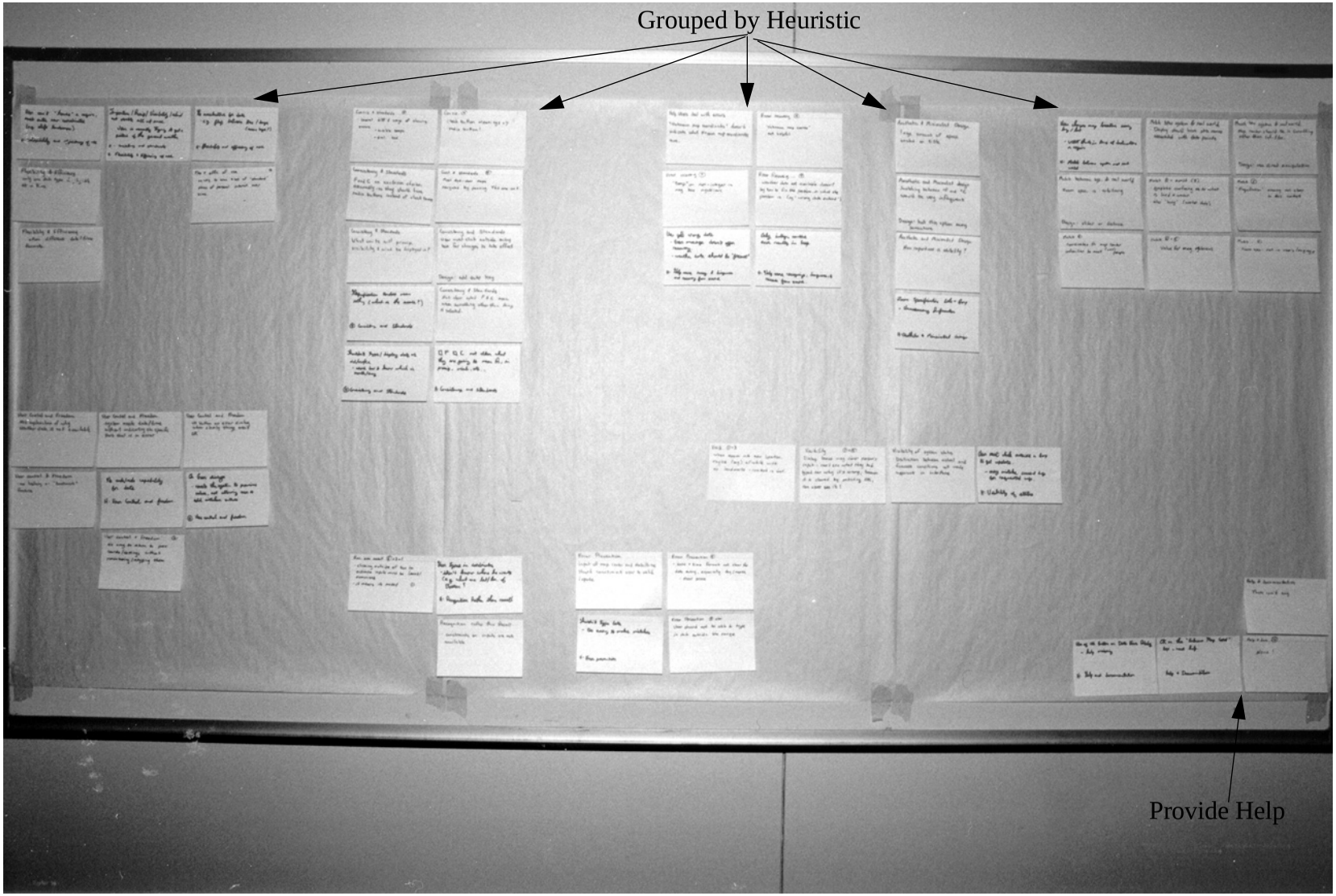


Figure 3.5: All cards laid out into groups by heuristic

according to the heuristics that they wrote on the cards. In cases of multiple heuristics, the author chooses one of the heuristics to be used for this initial categorization. The heuristics are used to create the initial groups because it seems reasonable to assume that the inspectors will classify similar problems similarly. It turns out that this assumption is only partially true: different inspectors will often label similar problems with differing heuristics. And further, they often see different parts of the same problem from different perspectives during their separate inspections, leading to problem descriptions that do not appear, on the surface, to be related and may be labelled with different heuristics. In the retrospective reflection provided by results synthesis, these seemingly separate problems can be identified as part of a larger whole. This is what makes results synthesis necessary and worthwhile, as will be discussed later.

3.3.4 Reviewing the data

In the process of putting their cards on the wall, the participants will often begin to notice and discuss the contributions of the others, particularly those that are closely related to their own. While informal discussion amongst the participants is expected and encouraged, it is important that in addition to this informal review of the other participants' contributions, each participant review the entire data set once it has all been laid out on the work surface. Without this comprehensive review, the participants may leap to premature conclusions based only on partial knowledge of the data set. In forcing the participants to review all the data before acting, I hope to combat people's natural confirmation bias (Wickens, 1992).

While the focus for each individual is on reviewing the contributions made by other participants, this review stage also helps them to be reminded of their own contributions, especially if a significant amount of time has passed between when they did the inspection and when they perform results synthesis.

During this stage the participants are expected to talk amongst themselves. They discuss the meanings of each other's contributions. One of the common causes of discussion is an attempt to interpret specific problem descriptions, as the descriptions can be difficult to understand for those who did not write them. The participants will also note similarities

between various problem descriptions. At this point, if two or more problem descriptions are found to be duplicates – describing in similar language what the contributors agree is the same thing – then the cards are stuck together to indicate that they are to be treated as a single entity. Only the most obvious duplicates are identified and clustered at this point. If more extensive reorganization is allowed, it will thwart the participants attempt to systematically review the entire data set. Excessive movement of the cards will lead to the participants losing track of what they have and have not reviewed. Therefore at this point the participants are conservative in identifying duplicates. Figure 3.6 shows the workspace at the end of the familiarization stage with the identified duplicates stuck together.

3.4 Emergence

Emergence, explained in Section 1.2.3, is the notion that fully formed, robust ideas and interpretations are not immediately obvious from raw data, but come about only after working with and discussing the data. The goal of the emergence phase is to create amongst the participants a single view of the interface problems that is complete, coherent, and concise. Coming into results synthesis, each of the participants will have his or her own particular view of the interface that will likely overlap only partially with those of the other participants. The object of results synthesis is to take these disparate views and from them create a unified view that is more encompassing and more insightful than the mere concatenation of the individual views.

Emergence is at once both familiar and mysterious. It is familiar because it is something most of us have done and do naturally. It seems to me to be a part of our pattern-seeking nature. It is mysterious because the concept is a new one and hard to describe in detail. In the rest of this section I attempt to give an understanding of how emergence plays out in results synthesis by taking a series of snapshots of the work surface as the process progresses.

3.4.5 Snapshot #1 - The first global reorganization

Following the conclusion of the familiarization stage, the participants embark on a radical reorganization of the workspace. At this point, the participants are not satisfied with the

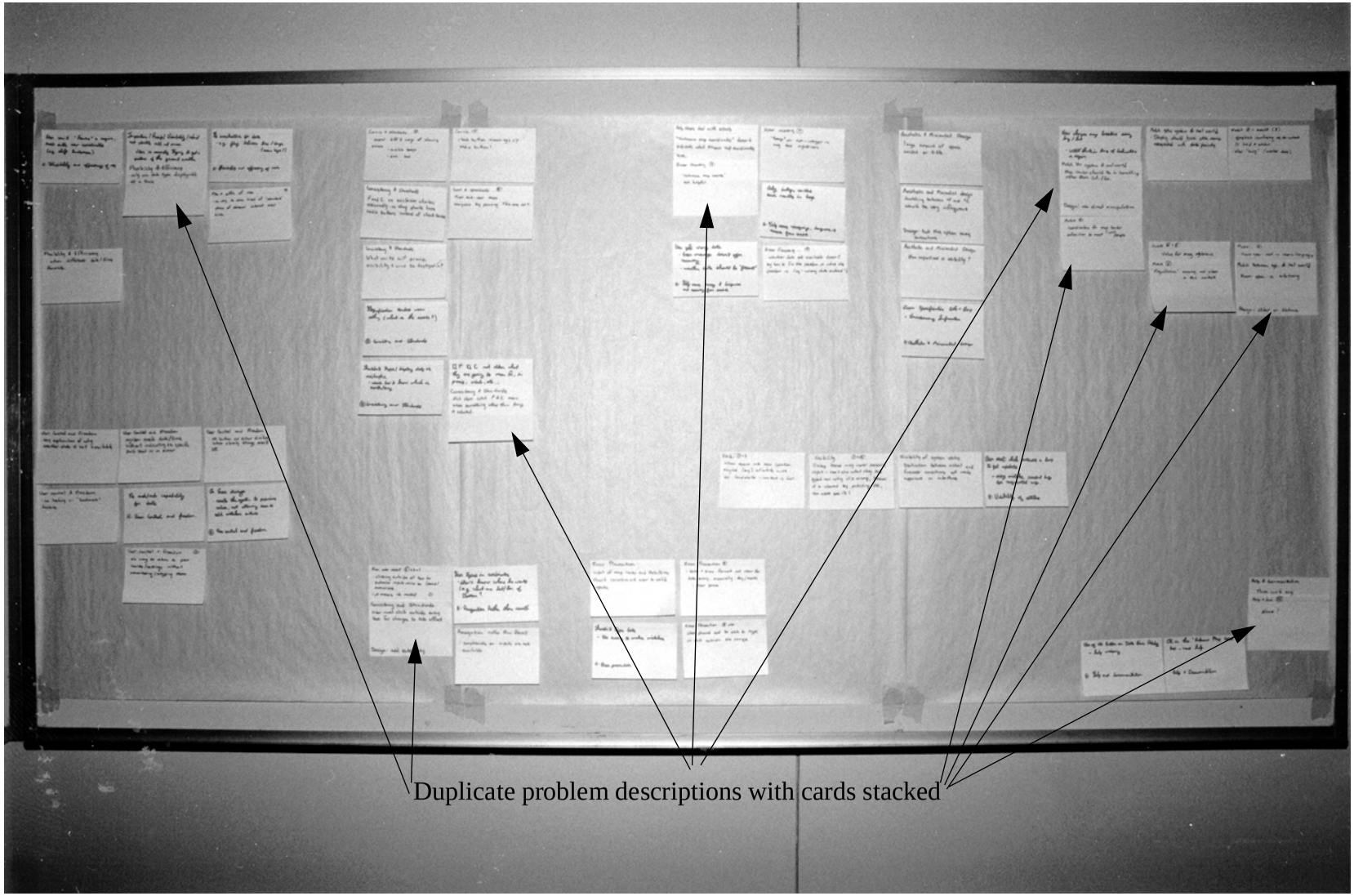


Figure 3.6: Workspace with duplicates identified at the end of familiarizing

existing layout of problem descriptions by heuristic. They have all noticed connections amongst cards that are widely separated. Problem descriptions that they now believe are symptoms of the same problem occur in a number of different groupings in the heuristic layout. All of the participants have expressed their dissatisfaction with the initial layout for this reason. The participants next try to come up with some universal principle that will allow them to reorganize the problem descriptions into more meaningful groupings.

One participant in particular pushes strongly for the data to be organized by “interface component.” Note that this and other terms have meaning to the inspectors with respect to the interface. What matters for this scenario is that the participants come up with new ways of organizing the data that are meaningful to them that were not previously present. These groupings emerge through the results synthesis process.

After some discussion, with frequent references to the work surface, the participants agree to a reorganization according to “interface component” (Figure 3.7). Towards the end of fashioning this layout, the participants begin to notice that they are not entirely comfortable with it. They begin to notice that not everything fits neatly into the scheme they are using. For example, in this layout the cluster in the middle-right dealing with “input” does not deal with a single part of the interface, but in fact three areas - “zoom/ mag,” “map center,” and “date.” Also, the lack of a clear separation between the “date” and “map center” clusters indicates the problems cannot be cleanly divided up using the chosen organizing principle. Again, I relate these terms not because they will be meaningful to the reader, but rather to show that the participants are thinking and expressing themselves in new ways that are very unlike the heuristics used to generate the raw data. There is no simple relationship between the groupings and the heuristics. These groupings emerge from the discussion and action around the emerging layout on the work surface.

3.4.6 Snapshot #2 - Organizing by problem begins

One of the lessons of emergence is that people should not expect to get it right the first time. It will take some time and some fiddling to get things into a satisfactory state. In this scenario, the group expects to try things that seem like a good idea at the start, but as they move along they realize that some of the ideas are untenable or undesirable. These are

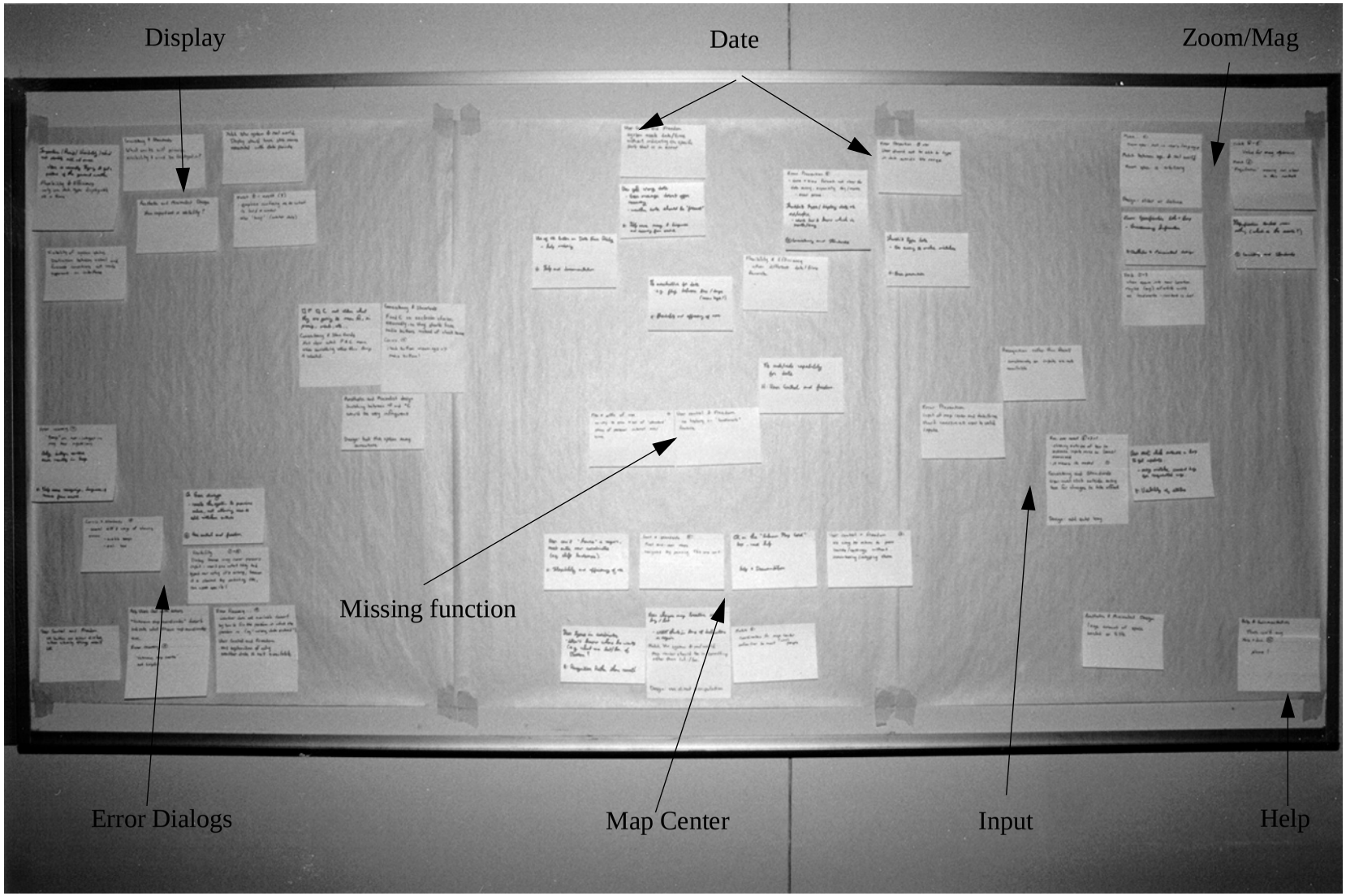


Figure 3.7: First global reorganization by interface component

abandoned in favour of new schemes that have been suggested by the working through of the original ones.

Having gotten most cards organized according to “interface component,” the participants then realize that this is not the best way to go either. This is not an epiphany, but a slowly growing realization. The realization grew with each instance of a data point that did not really fit well within the proposed classification scheme. The participants have now reached the point where they have exhausted their previous scheme without accounting for all the data. So they begin to cast about for a new way of looking at the problems in the interface that will allow them to account for and explain all the problems noted in the raw problem descriptions.

After more discussion, which again is carried out with much reference to the work surface, the participants begin to reorganize the cards again. Although they have not explicitly said as much, they are using what might appear to be an ad-hoc organizational strategy. In fact, they are seeking what Jeffries (1994) refers to as the right level of abstraction. The participants develop a notion of what best captures what they feel are the real problems in the interface. One may say that these are the *root causes* of the problems, but in fact the true roots probably lie much deeper. Rather, what the participants look for is a way of organizing the problems in the interface so that they will have maximum positive impact to the development of the interface. This depends, to a degree, on where the interface is in its development lifecycle, its market maturity, and the resources available to those who will be addressing the problems reported out of the Heuristic Evaluation.

Figure 3.8 shows the first three such groupings identified by the participants. In the lower right corner is a couple of cards labeled “No Help.” In the upper right is a group labeled “Options” with subcategories of “temp” and “date.” At the top in the middle in an emerging category titled “Use Geographic Names.” This process of identifying categories and arranging cards in them continues in an incremental fashion.

3.4.7 Snapshot #3 - Another problem area identified

The participants now adopt a strategy of moving through the workspace, identifying problem groupings as they go. This is a more collaborative activity, where all the participants

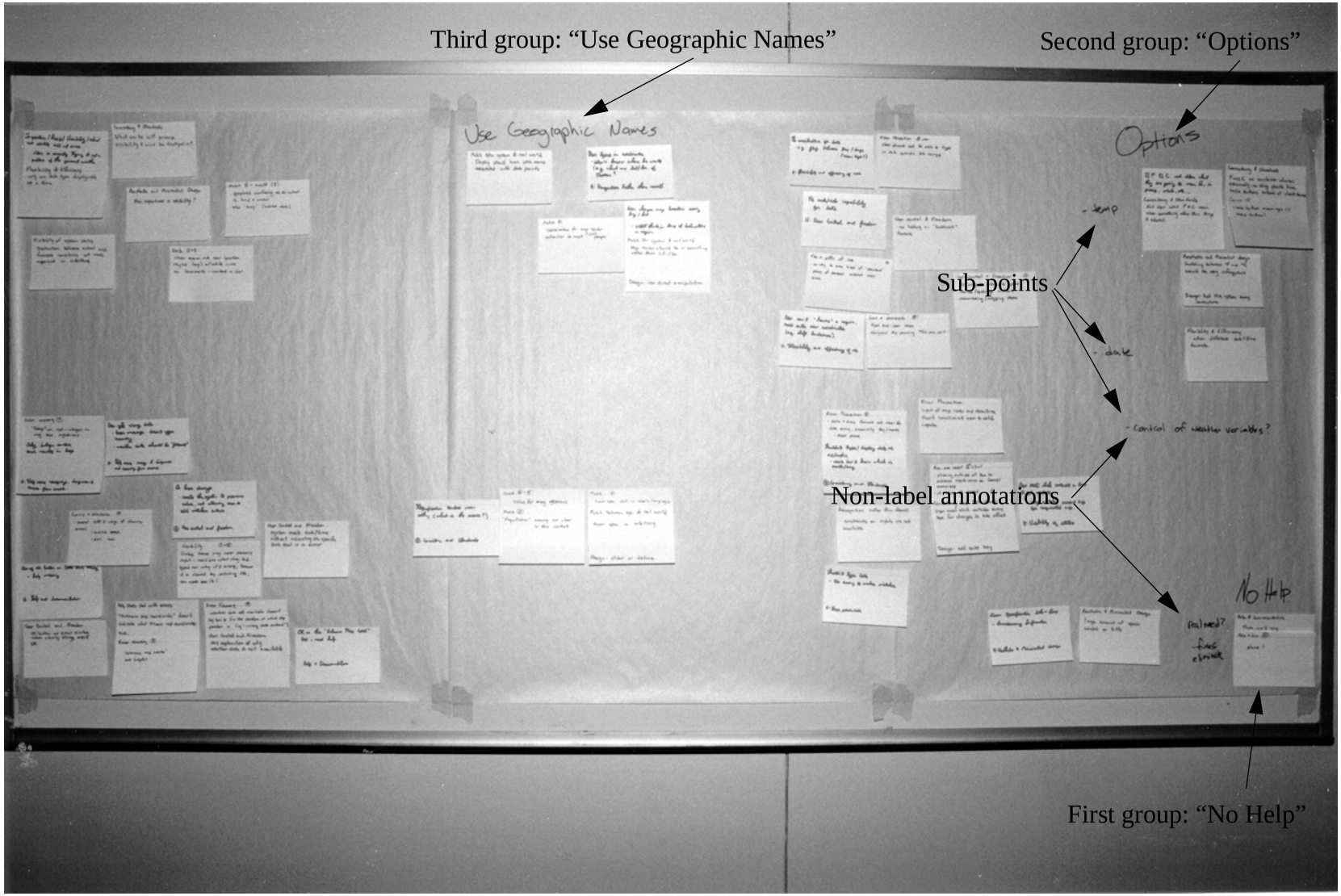


Figure 3.8: Second and third groups identified

will typically be attending to one grouping of the data in one part of the workspace. They will actively discuss the problem descriptions, trying to come to a consensus about whether the grouping is a single problem, what that problem is, and how the raw problem descriptions relate to each other as well as the final problem being proposed.

Figure 3.9 shows the state of the workspace after the next category has been identified. In the lower left corner a category entitled “Error Handling” has been created with three subcategories, each indicated by alignment and an annotation. Recall that in the first reorganization this area of the workspace was used for cards relating to the error dialogs (Figure 3.7), and that those cards have not moved much. However, the meaning of the area and the layouts within has changed dramatically for the participants. Even though what is physically on the workspace has not changed much, the participants understanding of what it represents has evolved dramatically.

Annotations are being used to carve up the workspace into areas that have been dealt with, and those that need further attention. In creating the annotation, the participants define an area that will contain all the problem descriptions relating to the problem that it identifies. Thus the annotations serve multiple purposes:

- the identification of a final problem;
- providing meaning to the spatial organization by connecting raw problem descriptions to the real problem;
- keeping track of progress and focussing the groups attention on the work left to do;

The creation of an annotation is the first step in making a final problem concrete.

3.4.8 Snapshot #4 - All but one

The participant proceed in identifying final problem groupings on an “easiest first” basis. Ease is not based upon the complexity of the problem or its proposed solution, but rather on agreement and confidence within the group that final problems have been identified. Thus, small, isolated, obvious problems are identified first, and the more ambiguous and interrelated issues are deferred until all else has been dealt with.

Figure 3.10 shows the workspace as it nears completion. Only the group of cards in the upper left remains to be dealt with. Of particular interest, note that the group in the center

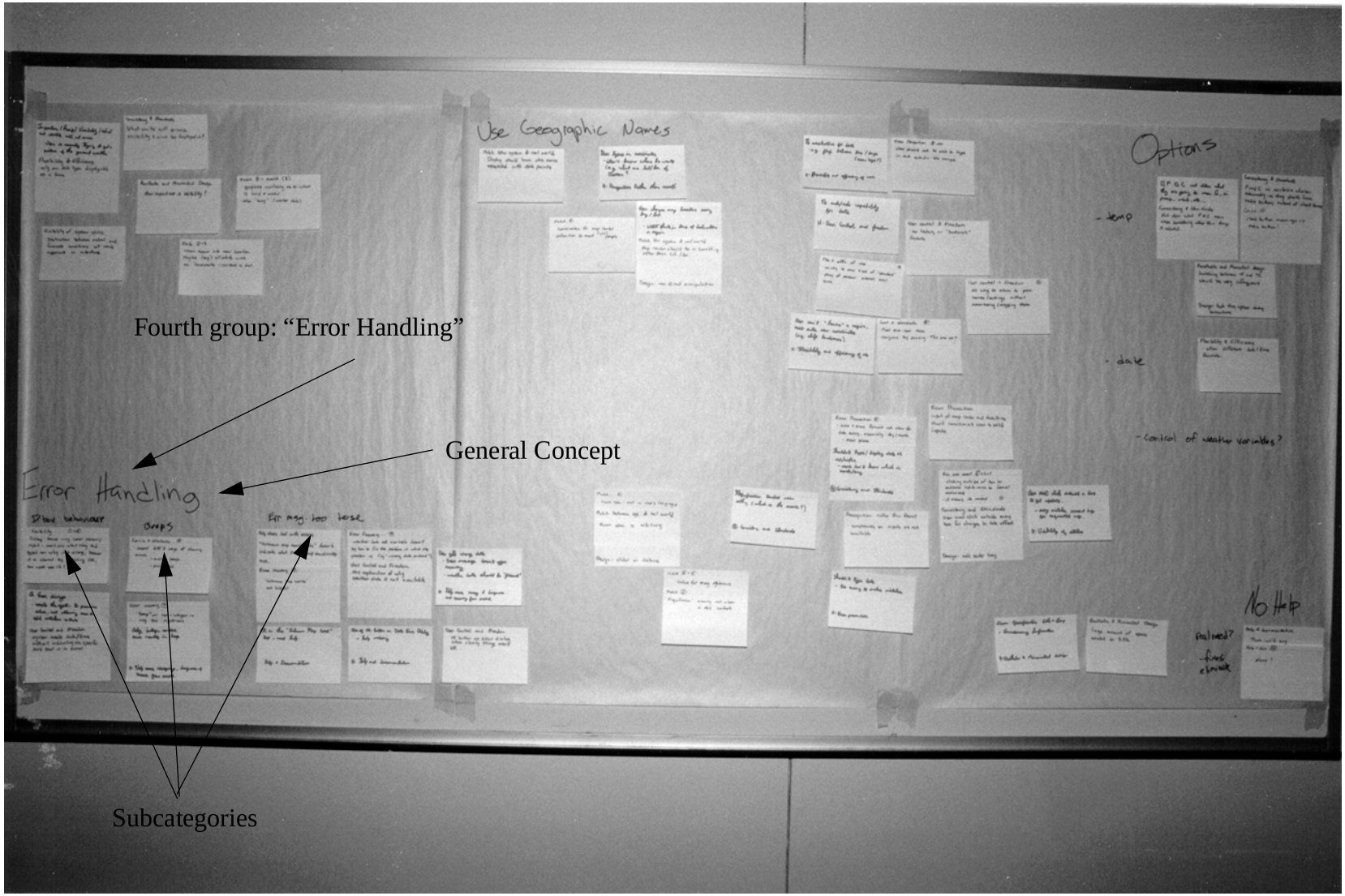


Figure 3.9: Fourth grouping identified

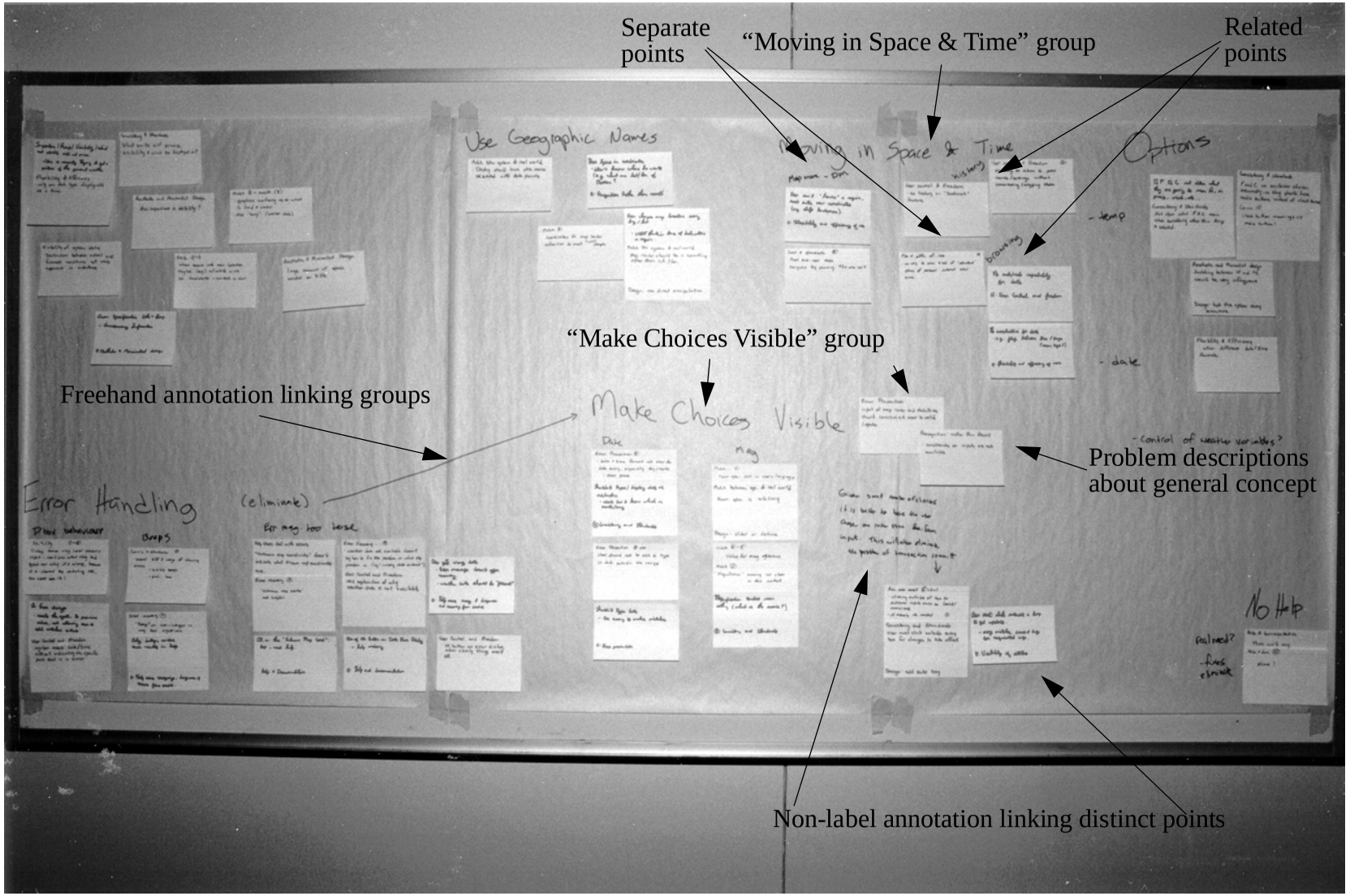


Figure 3.10: Workspace nearing the completion of emergence stage

bottom is related to the “Error Handling” group on the bottom left as indicated by an arrow annotation on the work surface. The bottom center group has a relatively complex organization and is titled “Make Choices Visible.” There is an extensive annotation that connects the main body of the group to three cards to the right. These three cards are the ones dealing with the way input is recognized in the existing interface.

3.4.9 Snapshot #5 - The final work surface.

The emergence phase completes once all the raw problem descriptions have been accounted for by a final problem group. The participants have come to a consensus about what the problems are in the interface, and how they are best expressed. This agreement and understanding is reflected in the workspace by annotations and spatial layout.

The final state of the work surface is shown in Figure 3.11. All of the cards have been grouped to capture what the participants feel is the best way of talking about and dealing with the problems in the interface. Only a few of these groupings relate to the original heuristics – most others define completely new ways of thinking about the problems.

3.5 Interpretation

The goal of the interpretation step is to take the understanding which is partially represented in the workspace and partially in the heads of the participants and to record it in a fashion that makes it available to those who did not participate. This is done by systematically examining the workspace and turning each group in the final layout into one or more problem reports. In this scenario the problem reports have three parts following the format recommended by Jeffries (1994):

1. *A detailed description of the problem in terms of users and tasks*.This description is to include how the user encounters the problem, what the user is trying to do, and why the problem arises.
2. *A description of the severity of the problem*. This describes how the problem will affect the users’ ability to achieve their goals, as well as other concerns such as their enjoyment of the work and appreciation of the software.

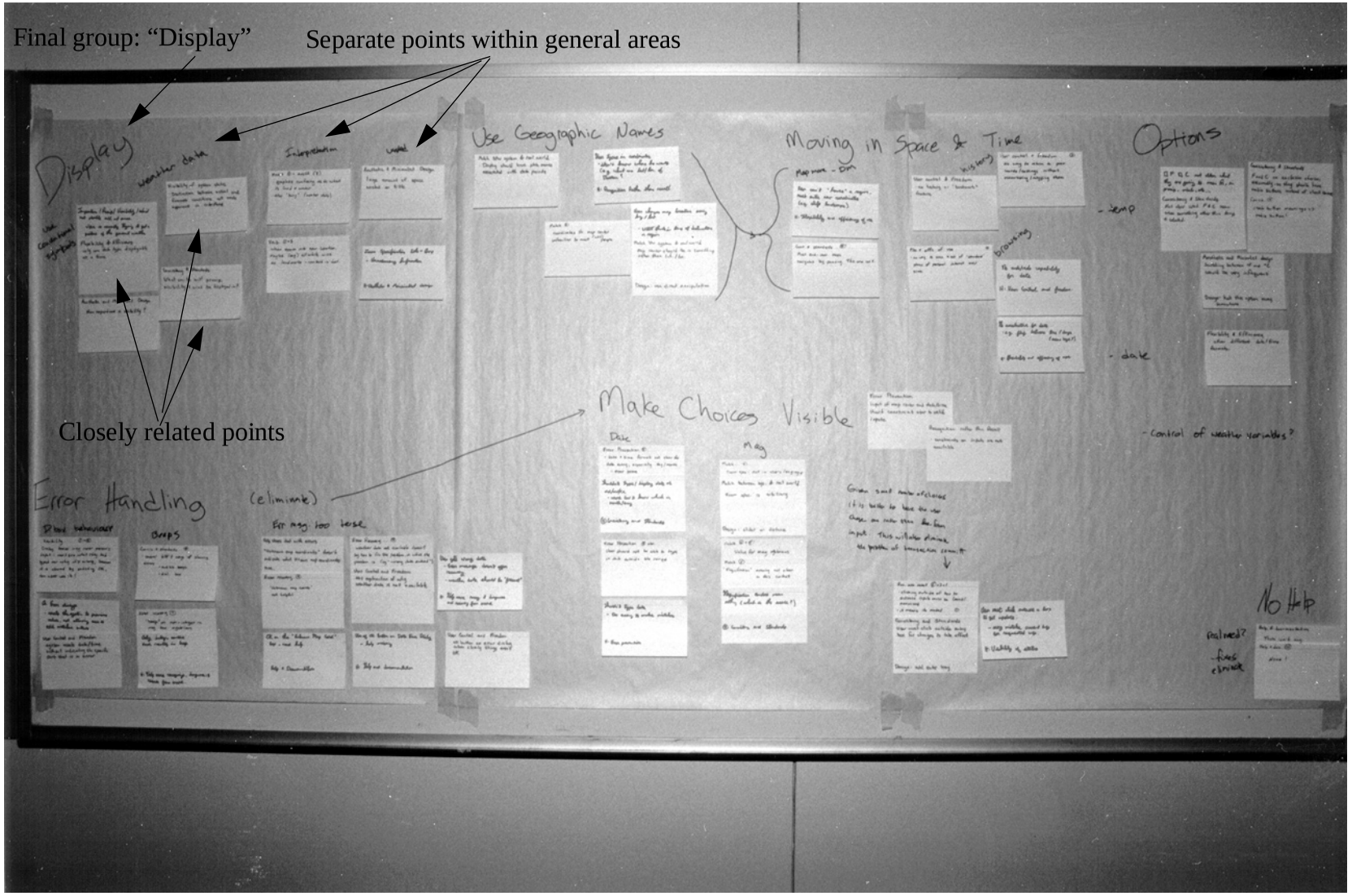


Figure 3.11: Final layout of workspace at the completion of emergence stage

3. *A description of the recommended action to take to remedy the problem*. This contains what the evaluators believe is the correct solution to the problem. If there is an alternate solution that is significantly less costly to implement but gives a substantial improvement, it too is included.

Figure 3.12 shows a grouping from the final layout in the scenario. This grouping is trans-

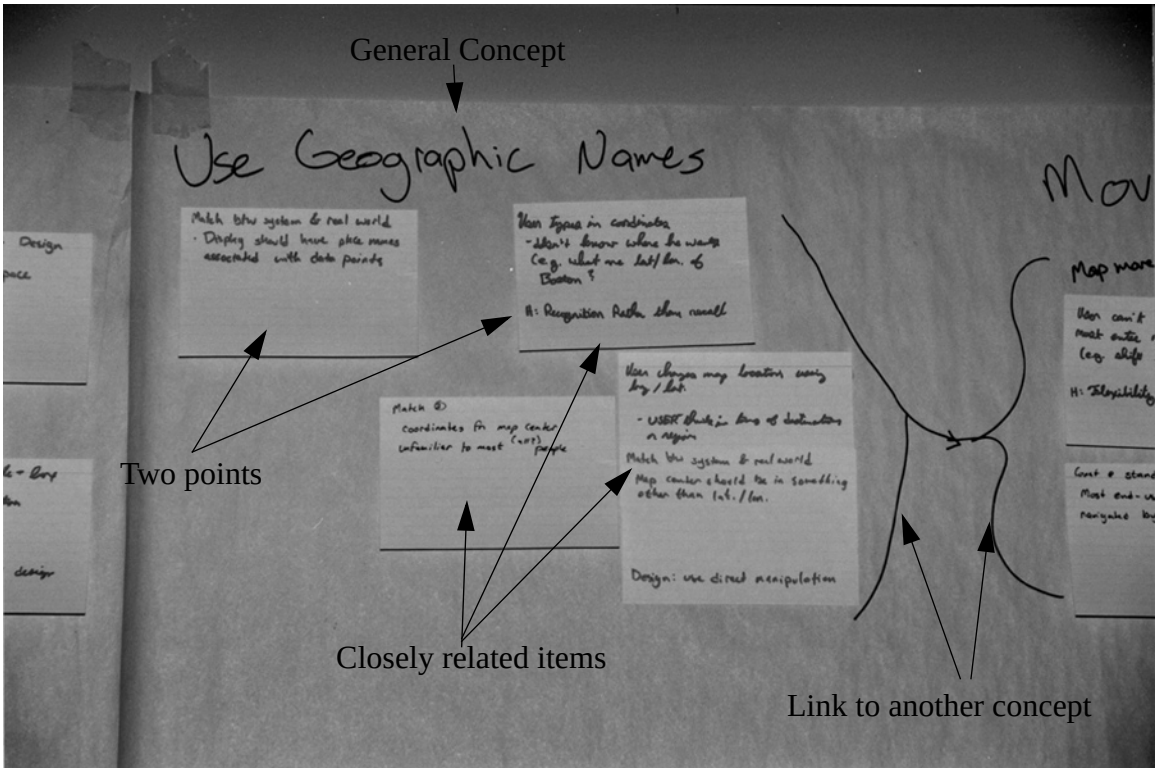


Figure 3.12: Close-up of "Use Geographic Names" layout in final organization

lated into the problem report shown in Figure 3.13, following the format outlined above. This demonstrates the large amount of information that the sparse layouts on the work surface actually represent. While what is in the workspace forms the basis of what is in the report, the report content is much richer than what is in the workspace as it is based on the understanding that emerged as a consequence of the participants going through results synthesis. Without the work done by the participants in the workspace, they would not have been in a position to generate such rich reports as their understanding would be much less developed. Appendix D contains a complete listing of problem reports for the layout shown in Figure 3.11.

Problem:

Most users of the interface will not be used to dealing with the world in terms of latitude and longitude. Rather they speak about the world in terms of geographic names of varying specificities. A user will be interested in the weather in the Maritimes, or southern Alberta, or Flin Flon. Most users will not have the resources necessary to convert these goals into the necessary latitude and longitude specifications. This will present an insurmountable barrier to using the system for many potential users. Those that do use the system are not likely to use it to its fullest extent because of the difficult of translating their desires into the appropriate actions. Also the lack of place names in the interface can make it very hard to understand what is currently being shown. Given the lack of geographic or political features shown in the display it is unlikely the users will be able to figure out what part of the world is being displayed and at what scale. To do so would force them to use a translation mechanism external to the interface to convert the latitude and longitude into a more familiar and sensible name. Further, forecasts are not given for a particular intersection of latitude and longitude in our experience, but rather for a named region – Prince Rupert, Labrador, or the prairies.

Severity:

This problem will affect both novice and experienced users in most tasks they would perform. The need to use latitude and longitude will be very intimidating to many novice users, some of whom may not even understand the term. While phenomenally motived users may be able to learn to use latitude and longitude, it is highly unlikely that they will be able to easily convert between that and the more commonly used names. Many are likely to rely on "cheat sheets" and not use the interface for getting information beyond a small set frequently used locations, perhaps only one. This constitutes a serious competitive disadvantage in our opinion.

Recommended Action:

Latitude and longitude provide a good mechanism for moving to random locations throughout the world or in regions that have no recognizable features or conventional names such as deserts or oceans. However, this is outweighed by the difficulty most users would have in translating between level at which they naturally express their intent and the level at which the interface is controlled. Therefore, control of the interface should be done primarily in terms of familiar geographic names. Further, rather than having type in names, which is error prone, the user should be presented with a means of selecting from among the valid choices. One mechanism would be a hierarchical list or browser. Another might be a pie menu scheme based on geographic proximity. By having the map center a named location, the need to have place names in the display area itself is reduced, given that the display is already going to be crowded with weather information.

Figure 3.13: Example problem report for "Use Geographic Names" grouping

3.6 Observational studies of groups performing results synthesis

The scenario I described in this chapter is not a baseless fabrication. In order to inform and substantiate my theoretical beliefs about what goes on in results synthesis, I performed four observational studies of groups carrying out results synthesis. I will first describe the studies and the substance of my observations in Section 3.6.1. In Section 3.6.2 I will discuss the ways in which the scenario I just presented differs from what I saw in observing real groups performing results synthesis in paper-based environments.

3.6.1 Observational studies

I performed four observational studies of groups engaged in results synthesis in paper-based environments in a variety of conditions which are summarized in Table 3.1. Where appropriate, informed consent was requested of and provided by subjects.

The first study had five participants of varied backgrounds, each a member of our research laboratory, and I was amongst the participants. The participants were:

- A male Computer Science professor who was familiar with my research on results synthesis
- A female with an M.Sc. in Computer Science and HCI training, then workting as a programmer
- Two males with M.Sc.'s in Computer Science and HCI training, then academic research associates.
- One male graduate student in Computer Science with HCI training (the experimentor).

The participants independently inspected an interface I took from Nielsen's *Usability Engineering* (See Appendix A). The resulting raw data produced from the inspection stage consisted of ninety-two problem descriptions, each labelled with a heuristic. The participants gathered in a seminar room with adequate whiteboard space and performed results synthesis on the generated problem descriptions. They spent about two hours on the task. While there was some contention about how the problems should be organized, the participants did produce a comprehensive and consensual organization. Due to a lack of time, the participants did not generate any problem reports, though they did arrive at a consensual final organization. During the process, spatial layout was used extensively. The end categories and concepts that emerged out of the work and discussion were very different from the original heuristics.

Study	# of participants	Participant type	Participant background	Interface evaluated	Interface maturity	Number of raw problem descriptions
1	5	Usability specialists and developers	Academic and commercial	Travel-weather	Screenshot and description	92
2	3	Developers	Academic	Own project	Working prototype	52
3	3	Developers	Academic	Own project	Working prototype	52
4	3	Usability specialists	Commercial	Prototype commercial product	Screenshots of working prototype	73

Table 3.1: Summary of observational study conditions

The second and third studies were of groups of three undergraduate computer science students who had been recently taught Heuristic Evaluation. The first group had three males, one of whom had considerable past experience with the task the interface was intended to support. All of the group members had been equaly involved in creating the prototype. The second group had two males and a female. One of the males and the female had implemented the prototype, the other male was responsible of other tasks. Each participant individually inspected an interface that the group of three was developing for one of their courses. By coincidence, each group generated fifty-two problem descriptions. As before, the groups gathered for results synthesis, but this time beginning with instruction in the general process of results synthesis. The process was facilitated, where the experimenter provided minimal direction if the group got stuck. In practice, this only happened at the transition between stages, if at all. Each study lasted for an hour and a half. In both cases there was substantial movement of problem descriptions in the workspace. The heuristics were not used beyond initial layout. The participants made extensive use of spatial layout. They ran out of time before generating problem reports, but both groups considered the exercise to be valuable, learning things about the interface, its problems, and their potential solutions. One of the groups requested to be allowed to continue the activity past the end of the study.

The fourth study was of three practitioners from industry performing results synthesis on a early product interface they had inspected. Two of the participants were males, and one was a female. One of the male participants had a Masters degree in Human Factors, had been involved in the early design of the prototype, and was the leader of the Human Factors group. This organizational power was not observed to inhibit the other participants in expressing their opinions nor did I observe this participant to over-rule the other participants. The second male had a Bachelor’s degree in Mechanical Engineering and had been involved in the design of previous prototypes of the system. The female had a Bachelor’s degree in System Engineering and had not been involved in the design of the system, though she was aware of the system and its general goals. She also was not formally a member of the Human Factors group, but had HCI training. All of the participants were

associated with the Human Factors group of the company developing the system. The inspection and anything that came out the exercise would be of benefit to them in their work, though the human factors lead for that particular product did not participate. The participants generated seventy-two problem descriptions. Results synthesis unfolded in much the same way as seen with other groups. During the process there was much movement of the problem descriptions about the workspace. New categories emerged as the group considered and discussed the collected data. Once again, spatial layout was used. The study lasted for an hour and a half, at which time the participants had started to formulate final problem reports. As with the other studies, the participants ran out of time and did not complete the task.

What I confirmed as a result of these studies was:

1. Groups can perform results synthesis. I watched groups of varying sizes with widely differing compositions carry out the process with reasonable efficiency, making significant progress in the short time allotted.
2. Results synthesis is seen as a useful thing by participants. In post-study interviews, the participants indicating that as a result of performing the process they learned more about the interface and its problems. They also appreciated how it encouraged valuable team interaction and sharing of knowledge and perspectives that would not have otherwise occurred.
3. Results synthesis is a “natural” activity. The participants were provided with only a high level description of the process, yet were able to carry out the activity with only minimal facilitation from the experimenter; in practice, facilitation was needed only at stage boundaries.
4. Spatial environments are good for results synthesis. All of the groups I observed took advantage of the affordances of spatial environments for representation of ambiguity and subtle gradations of relationships amongst the problem descriptions.
5. Emergence occurs in results synthesis. The final arrangement of elements within the workspace, and what this meant as problems in the workspace, was not predictable or

directly derivable from the starting condition. Rather, the final arrangement results from the participants interacting with one-another and the workspace.

6. The participants knew more about the interface, its behaviour, the target users and their tasks, and understood more deeply the problems in the interface as well as the design space and trade-offs made in the current design, than when they started. In the post task interview, the participants indicated that they had benefited from the process in terms of noticing new problems, new relations, and new perspectives.
7. Problem descriptions can be too cryptic to be interpreted by anyone other than the author. I observed participants verbally indicating their inability to interpret problem descriptions and requesting clarification from the author about what part of the interface was being referred to and what the author thought was problematic about it.
8. Duplicates occur, and can be non-trivial to identify. In all cases, the inspectors independently produced problem descriptions that referred to the same problem in the interface, though these were often not labeled with the same heuristic and occasionally were not expressed in similar phraseology.

3.6.2 Differences between the scenario and my observations

Though I believe the above scenario captures the common and expected behaviours, the results synthesis sessions I observed differed from the scenario in a number of ways. The observed sessions also differed one from each other – no doubt in part because of different participants, and different types of participants.

One of the differences was in the use of annotations. In this scenario, annotations are used extensively. In the observed sessions, the first group studied used annotations extensively, including uses other than labeling of problem description groups. Non-label annotations may be a feature of bridging between the emergence and interpretation stages, helping the participants to record their understanding and stabilize the meaning of otherwise potentially underdetermined spatial layouts (Marshall and Shipman, 1995). Similarly, the group observed in the fourth session used non-label annotation, though not as extensively as the first group. In the middle two sessions, the participants used annotation sparingly, and only as area labels. This may be in part due to time restrictions, though no

doubt other factors were at play, such as the relationship between the evaluators and developers, and the experience of the participants and groups.

Another of the differences is that in this scenario, the boundaries between the organizing phases is more distinct than in the observed sessions. Also, the reorganization is more drawn out in this scenario. In all the observed sessions, there was one big reorganization, namely the change from grouping by heuristic to discovered groupings. These second groupings were discovered by the participants by noticing commonalities in the problem descriptions. The grouping would be suggested to the other participants, they would discuss it, and if it was approved, the group would be created by a label annotation and search of the workspace for members. This process was incremental, much like the second phase of reorganization in this scenario. In the observed sessions, there was also a different mix between macro-level organizing (locating group members and moving them to the same area) and micro-level organizing (arranging members within their area to reflect their perceived relationships). In this scenario, the distinction between the two levels of organizing are not well drawn out, and very little micro-level organizing is apparent. This is partially due to the difficulty in describing a highly fluid process in text and a few pictures.

This scenario posits a multidisciplinary group of participants. The first session was the most multidisciplinary, with the other three being more homogeneous – usability practitioners in the fourth session, and inexperienced developers in the second and third.

3.7 Conclusion

In this chapter I have presented a scenario describing how results synthesis may be carried out using paper as the working media. There are four stages in the scenario. In the preparation stage the participants write problem descriptions on index cards, and gather in a room with adequate wall space to organize them. The familiarization stage comes next, where the participants put all the cards, one per problem, on the work surface, and review the entire collection. The third and most important stage is emergence, where the participants rearrange the cards upon the work surface to reflect their emerging understanding of what is wrong with the interface and how to best represent it. The final stage is interpretation wherein the participants convert their understanding, based on the workspace, into a more

conventional text form to facilitate feedback to the developers of the interface evaluated. I also compared the scenario with observations of actual groups performing results synthesis, where the diverse groups did engage in the expected behaviours, essentially congruent to what was described in the scenario. In the next chapter I will present detailed requirements for supporting results synthesis.

Chapter 4: Requirements

4.1 Introduction

Results synthesis is the transformation of the problems found in the inspection phase of Heuristic Evaluation into an outcome that leads to development giving appropriate consideration to the problems discovered. In this chapter I present the requirements that I believe are necessary for effective results synthesis. These requirements are derived mostly from the Heuristic Evaluation literature, as well as that on usability inspections and usability evaluation in general. The exceptions are the requirements in Section 4.4 which are based on research by Cathy Marshall and her colleagues into how to support emergent structure.

In this chapter I have grouped the requirements into five categories for presentation purposes. The first group of requirements (Section 4.2) concerns the necessary elements to be included in problem descriptions from the inspection stage. These raw problem descriptions do not have the necessary quality to be passed directly to development (Jeffries, 1994), so they must be refined. Thus the second group of requirements (Section 4.3) aims to ensure that the outcome of results synthesis will have the necessary qualities to be effective. I have further categorized these qualities as completeness, coherence, and conciseness. Many of the qualities can be obtained only after substantial processing of the raw problem descriptions. The third group of requirements (Section 4.4) enables this process to be carried out in an effective manner. I argue that the most effective manner for people to carry out the processing is for them to use spatial organization techniques to organize and transform the raw problem descriptions. The form in which the final problem reports are recorded is the subject of the fourth group of requirements (Section 4.5). The final problem reports must be documented, must separate the description of the each problem from the description of its solution, and must describe the severity of each problem. The last group of requirements (Section 4.6) deals with who performs the results synthesis. I argue for a participatory approach to results synthesis where the inspectors of the interface as well as its developers and end users are included in the process.

4.2 Elements of raw problem descriptions

A prerequisite of results synthesis is that the evaluators have already performed the inspection stage of Heuristic Evaluation and during the inspection recorded information about the problems they found. Nielsen (1994b) stated that for each problem found during the inspection phase the inspector should record a description of the problem, the associated heuristic, and any solution to the problem the evaluator may think of at the moment. Problem reports that include these three elements, plus an identification of the inspector who recorded the problem, provide the raw material that serves as the input into results synthesis.

The requirements I am proposing below do not make significant demands beyond what is already recommend by Nielsen (1994b). The identification of the evaluator is, I believe, implicit in Nielsen's (1994b) description of Heuristic Evaluation.

Requirement 1: Include a description of the problem.

Each problem found is described in terms of users and tasks (Jeffries, 1994). The level of detail necessary in the description will depend on how results synthesis is being performed

Requirement 2: Include the associated heuristic.

Each problem found is annotated with the relevant heuristic from the set being used in the inspection phase. For readers of the raw problem description the heuristic functions as a shorthand justification for the problem being reported. For the author of the raw problem description, it provides a cue to help recall what he or she was thinking when the problem was "found."

Requirement 3: Provide the inspectors with a space to record possible solutions.

Solutions are recorded in the raw problem descriptions to allow the inspectors to capture any insights they had in the process of discovering the problem. It also serves to remind the inspectors that their problem descriptions should not be phrased as solutions.

Requirement 4: Include an identification of the author.

Each problem report allows for the identification of the inspector who authored that particular raw problem report. The benefit of having an identification is the ability to get an explanation, should the raw problem report not be clear enough.

While all raw problem descriptions have the above mentioned components, there is no

Simple and Natural Dialogue
If there is room, you should write out the entire word instead of using abbreviations. Thus, "October" is preferable over "Oct."

Figure 4.1: Simple Heuristic Evaluation problem report after Molich and Nielsen (1990)

Problem: Some dialog boxes have the labels "Apply," "Close," and "Cancel," while others use "OK" and "Cancel." This will be confusing to users, who in both situations are looking to commit changes they have made. While the choice of a button set appears to be based on an analysis of the task being done—the Apply set seems to be used in cases where the user might want to try out several examples before dismissing the dialog box, whereas OK is used when a one-time operation is likely—new users may not be sensitive to those distinctions and may be confused when they encounter an Apply dialog box, which is the less common version.
Severity: Users will encounter this discrepancy regularly. Once (if) they infer a justification for it, the difference may not negatively impact them; but until they do, they will often be looking for a button label that is not present, which will slow them down each time they encounter a dialog box. It will have the biggest impact on new users, but many users may never infer the rationale that makes the different options less arbitrary.
Solution: Because the lack of consistency is so jarring and shows up in so many places, I would recommend using the same option in all cases—the Apply/Close/Cancel set, since it provides the more flexibility to the user, albeit at the cost of an extra button press (but it saves having to invoke the dialog box a second time, which is a more complex operation).
An alternative solution that improves consistency somewhat while keeping the task-specific approaches would be to label the OK button as Apply+Close. That makes clear the relationship between the button sets, makes the labels more consistent, and still provides tasks specific functionality. New users will still find this jarring, but many more of them will be able to infer the relationship between the two button sets and be able to find the button they require (usually, the Apply button) based on its label.

Figure 4.2: Jeffries' (1994) example improved problem report

agreed upon level of detail that should be provided in the reports. The problem reports listed in Nielsen's publications about Heuristic Evaluation are short and terse (Figure 4.1). In contrast, Jeffries (1994) describes an alternate standard for problem reports that has much more detail (Figure 4.2). Jeffries argues that this increased level of detail is necessary to avoid the sorts of problems she observes in an attempt to perform results synthesis on the raw problem descriptions generated as part of an earlier study done by herself and others (Jeffries et al, 1991). Her goal is to prevent problem reports from being misunderstood. To this end, she suggested:

- Descriptions of the problem be distinct from the descriptions of the solution
- Problems and solution both be explicitly justified
- Problem severities be described in terms of users and tasks
- Design trade-offs affecting the problem and solution be explicitly identified and explored.

Raw problem descriptions intended for results synthesis, as construed in this research, should be as short as practical, such as those is Figure 4.1. The trade-off between terseness and detail in the raw problem descriptions is made on the basis of the amount of work required to write the problem report to the level of detail required. Extra work in recording the problem means less time for finding new problems. Another factor to consider is the amount of time it takes to read and comprehend the problem descriptions. When there are fifty or more problem descriptions, the difference between reading a few sentence fragments or three substantial paragraphs is significant. The level of detail and precision recommended by Jeffries may be appropriate in final problem reports that are the product of results synthesis (See Section 4.5), or in other situations where misunderstanding of the problem reports must be minimized and the authors are not available for consultation. A results synthesis process that meets the requirements laid out in the rest of this chapter needs only the much simpler style of raw problem report such as that shown in Figure 4.1.

4.3 Ensuring a quality outcome

There are a number of goals to accomplish when transforming the raw problem descriptions from the opinions of individual inspectors into an outcome that can be communicated effectively to development. In this section I present these goals as requirements intended to ensure that the necessary actions are taken and that the necessary qualities are present in the outcome. I derive these requirements from the observations of other researchers who have reported about various properties of evaluation results that have had effective impact with development.

I have grouped the requirements into three categories: completeness, coherence, and conciseness.

4.3.1 Completeness

In the inspection phase of Heuristic Evaluation, the emphasis is on finding problems. This leads to raw problem descriptions that sketch out the problem and the surrounding concerns. The requirements I have brought together under the category of completeness encourage a deeper consideration of the problems that are to be presented in the outcome. This consideration is to include the context for the problem, the justification of the problem, and what action is to be taken to fix the problem.

Requirement 5: Consider the entire collection of raw problem descriptions.

It is only through the active consideration of all the raw problem descriptions that results synthesis will be effective (Jeffries, 1994). Every problem is considered with respect to all the others. Those performing results synthesis should avoid dealing with the raw problem descriptions as isolated individual issues, but rather try to discover the relationships between each problem described and all the others that were generated in the inspection phase. If problems are considered in isolation, important commonalities or contradictions may be missed, resulting in a flawed outcome. In the case that results synthesis in being performed by a group, the participants should avoid dividing the raw problems amongst themselves and working independently on a subset of the entire collection. By narrowly focussing their attention, they are likely to obscure the connections that may exist in problem descriptions that at first glance seem unrelated (Jeffries, 1994).

Requirement 6: Consider all the operative constraints on the interface.

Those performing results synthesis must be familiar with both the goal of the interface and its relationship to other applications if they are to avoid recording problems or solutions in the outcome that are false alarms. Jeffries (1994) defines false alarms as recommendations that, if acted upon, would have no usability benefit and may even reduce the usability of the interface. False alarms are often due to misunderstanding, either by developers of what usability is trying to say, or by usability specialists of what the constraints are on the design of the interface. Many of the other requirements presented in this chapter are aimed at eliminating the first sort of misunderstanding. This requirement is aimed at eliminating the second sort of misunderstanding. When

the inspectors are not familiar with the goals and constraints of the interface they are liable to find problems and solutions that are contrary to the intent of the interface, or that are in conflict with what is done in the associated applications. An example of the sort of false alarm that is to be eliminated by this requirement is a recommendation that would make keyboard accelerators consistent with a popular application, but inconsistent with the suite of other applications with which the interface being evaluated is intended to work.

Requirement 7: Provide adequate justification for every problem.

Adequate justification is a compelling explanation of why the problem will exist for the end users of the interface being evaluated. The level of detail necessary to provide adequate justification will depend on the organizational context in which results synthesis is performed. If the evaluators and developers of the interface are suitably like-minded, then the mere mention of a heuristic may be adequate justification. However, this situation may not be commonplace, so adequate justification is usually a reference to the applicable standards or guidelines for the interface or argumentation about why end users performing their tasks using the interface will experience the problem. One of the important goals of providing justification is to forestall the objection by development that the problems raised by the Heuristic Evaluation are mere personal opinion and need not be addressed, as the next “expert” to look at the interface will likely provide a different and possibly contradictory opinion. The aim of providing justifications is to show that the problems are not based on “What I like” but accepted standards and practices of all usability professionals.

Requirement 8: Consider all the applicable trade-offs in the interface.

One of the primary functions of design as an activity is the making of trade-offs between all the competing and often directly opposed demands on the artifact being designed. Though the designers and implementers of interfaces for software applications are working in a very malleable medium, they are still faced with many trade-offs. They make these trade-offs based on their beliefs about the intended users of the interfaces and the tasks the users will be performing, as well as other constraints on the

development of the application. The evaluators may see problems in the interface that are the result of differing beliefs about the best way to make the trade-offs. Including problems and solutions in the outcome that do not acknowledge the trade-off being made can lead to developers dismissing the recommendation because they will not see why the trade-off they made is not satisfactory. Thus, by acknowledging the trade-off and arguing for a different way of making it, the developers will be forced to think about the different ways of making the trade-off and will not be able to dismiss the problem and proposed solution out-of-hand. Thus, consideration of problems and solutions must include an acknowledgment of the design space in which they exist, and the position of both the existing interface and the proposed change within that space.

Requirement 9: Generate workable solutions to the problems raised.

Developers are often not in a position to come up with solutions to the problems that are raised by Heuristic Evaluation as they may not have the necessary experience or training to formulate a solution. The participants in results synthesis should have the experience and training necessary to create workable solutions. What constitutes a workable solution will depend on the skills of the developers, the skills of those involved in results synthesis, and where the interface is in the development lifecycle.

Some may argue that engineering quality assurance only reports the problems discovered, and so usability engineering evaluation activities should aim to operate on the same level. These two activities are fundamentally different. Engineering QA can been seen as ensuring that the developers have done what they said they were going to do. It tests the behaviour of the application. Usability engineering evaluations, on the other hand, test the behaviour of the application as it interacts with the behaviour of the end user. The problems found in usability engineering are a result of the designers and implementers not foreseeing the way in which the user-interface interaction would play out, rather than the developers failing to do what they said they were going to do, which is the case with problems reported by engineering QA. Thus it is enough for engineering QA to report the problem found as the developers are in principle in possession of all the information necessary to program the desired behaviour. In contrast,

usability engineering evaluations should report a solution to the problems found as the evaluators are the ones most likely to understand why the interaction between program and user behaviours went wrong and how to change the system's behaviour in a way that will eliminate this problem and not cause others. Developers may not have this understanding, or the information which underlies it.

Requirement 10: Generate alternate solutions for problems where possible.

If the solution to a problem is expensive, then alternative solutions that are less costly should be included, even if they are less than ideal (Sawyer, Flanders, & Wixon, 1996). This allows development more flexibility in their making their resource allocation trade-offs and shows that usability is sensitive to the realities of software development.

4.3.2 Coherence

The inspection phase of Heuristic Evaluation produces a set of problem descriptions that are clearly the work of several individuals. In order for the outcome of results synthesis to be effective, it must be presented as a unified whole. Thus by coherence, I mean that the outcome of results synthesis be free of apparent self-contradiction, and that it speaks with a single voice. The requirements in this category are intended to promote clear communication of a single agenda to the developers.

Requirement 11: Create a consistent set of underlying assumptions.

The more disparate the group that performs the inspection of the interface, the more disparate their assumptions about the interface. These differing assumptions are reflected in the problems discovered, the severity attributed (Jeffries, 1994), and the solutions considered. The outcome of results synthesis is enhanced by making these assumptions as explicit as possible, and then having a single set of assumptions about users, tasks, the goals of the interface, and the priorities of problems expressed in the problem reports which compose the outcome. The more inconsistent the assumptions, the greater the danger of reporting contradictory problems or solutions, and the greater the danger of results synthesis being derailed by debate over issues traceable to differing assumptions. One cue to differing assumptions are problem descriptions that request additional functionality (Jeffries, 1994). Such problem descriptions deserve

extra scrutiny, as they are often the result of differing assumptions between the designers and the evaluator. The evaluator is assuming that the interface is intended for a purpose that requires the functionality he or she is requesting. The interface may have been designed for an entirely different reason for which that functionality is unnecessary.

Requirement 12: Create consistent language and terminology.

Raw problem descriptions of the same problem often use very different ways of describing the problem (Jeffries, 1994). Converging to a single way of expressing the problems helps to identify duplicate raw problem descriptions and unify assumptions.

4.3.3 Conciseness

The outcome should have as few problem reports as possible to cover all the problems in the interface. Refining the outcome to the most compact expression possible promotes clarity, both in those producing the outcome, and in those using it. When the outcome of results synthesis is concise, it clearly expresses what development should do. This enables development to confidently make the resource allocation decisions necessary to successful product development.

Requirement 13: Eliminate duplicate problem reports.

It is almost a given that the evaluators will find some of the same problems (Nielsen, 1994b). Thus, a number of the raw problem descriptions will be redundant. Because of differing terminology, phrasing, and levels of abstraction in the problem descriptions, duplicates may not be immediately obvious (Jeffries, 1994). Therefore, a specific effort is needed to ensure that duplicates are removed. All of the raw problem descriptions of a single problem should be replaced by a single representative. Leaving duplicates in the outcome will likely cause developers to doubt the thoroughness of the evaluation, and is a sign of disrespect towards the time pressures which developers are under.

Requirement 14: Express problems and their solutions at the right level of abstraction.

Raw problem descriptions are typically symptoms of broader or deeper problems (Jeffries, 1994). If the problem described in a raw problem report is found in a number of

different locations in the interface, then the corresponding final problem report should describe a problem with the entire system that appears in a number of instances and the solution should aim to fix all of the instances at once. Developers are unlikely to have the time, ability, and interest to consider each problem for its wider implications. If developers are given a problem that cites only a specific occurrence, they may fix only that occurrence and cannot be faulted for failing to generalize the problem. Those performing results synthesis have the ability and the mandate to spend the time necessary to find the level of abstraction at which problems are easiest to describe and fix.

Requirement 15: Examine ambiguous problems.

In the process of producing a justification for a particular problem (Requirement 7) those performing the results synthesis may come to believe that they do not have sufficient knowledge of the end users for the interface and their tasks to decide whether or not the problem will be discernible by the end users or have any measurable impact on their task performance. Sawyer, Flanders, and Wixon (1996) refer to such problems as "ambiguous" and suggest that they be include in the outcome. I believe discussion of such problems is worthwhile, and if Requirement 25 on including end users is being met, it is much more likely that problems that are ambiguous for the evaluators and developers will be resolved. Otherwise, a decision will have to be made whether to include these problem reports in the outcome. Including them may be valuable for keeping a record of what further information about the end users and their tasks needs to be gathered.

4.4 Enabling emergence in results synthesis

One of the long-standing projects in human-computer interaction and related fields has been the creation of systems that are intended to help people think and express themselves more effectively (Englebart, 1968). However, these systems have had at best mixed success. Some researchers have attributed this to the system designers focussing on supporting the finished product, rather than the process by which it is achieved (Marshall & Shipman, 1995).

The authoring process is one that has been recently studied with the intent of how it may be best supported (Monty, 1990; Marshall & Rodgers, 1992). The major finding of these studies is that the final conceptual structure did not exist at the beginning of the activity, but rather emerged as a result of the authors' changing understanding of what was important and how to best express this understanding. This concept of emergence is important in a wide range of creative activities (Shipman & Marshall, 1994) which may be categorized as being essentially design activities (Edwards, Moran, & Do, 1998).

Results synthesis is one such creative activity where emergence phenomena are central to the process. Many of the requirements in Section 4.3 cannot be met by simple arrangement of the raw problem descriptions, but can only be satisfied by extensive processing. In what is essentially a case study in results synthesis, Jeffries (1994) reports that duplication and contradiction between raw problem descriptions is often not readily apparent — that it emerges only after extensive work with the problem reports. And certainly something as seemingly nebulous as "the right level of abstraction" will not simply fall out of the raw problem descriptions, but will only emerge after working with the whole collection of problem reports over enough time to get a satisfactory understanding of all the problems and their surrounding issues. This understanding itself is something that emerges only from dealing with the entire collection of problem reports over the course of results synthesis.

Monty's (1990) study of the authoring process shows that it has many parallels to the results synthesis:

1. The author in Monty's study dealt with information coming from many different sources which contain points that may be similar, complementary, or contradictory. Results synthesis deals with raw problem descriptions coming from different evaluators that may identify the same problem (Nielsen, 1992), different problems (Nielsen, 1992), or contradictory problems (Jeffries, 1994).
2. The author had to deal with a large number of source notes, and results synthesis may deal with similar numbers of problem descriptions (Muller et al, 1995).

3. The author must establish a single way of talking about his subject, and must establish a translation between his chosen language and that used in all of his sources. Similarly, results synthesis must establish a common language for describing problems in the interface in order to meet the requirements of Section 4.3 as the problems in the raw problem descriptions are often described in very different language (Jeffries, 1994).

Results synthesis is a form of authorship, and as such, the findings of Monty's (1990) study and that of subsequent researchers studying similar activities apply to results synthesis.

The researchers on whose work I have based these requirements were focussing on building systems to support people engaged in activities that have emergent features. Thus, they often phrase their findings in system specific ways, even though they have broader implications. The requirements I present in this section apply equally to systems supporting results synthesis as well as physical media that may be used to support results synthesis such as Post-It™ notes and whiteboards. Indeed, Monty (1990) identifies index cards as a media that supports authorship better in many regards than the existing software systems of the day.

Requirement 16: Provide a spatial environment.

An everyday example of a spatial environment that supports results synthesis is any large flat surface such as a wall. The wall provides a surface on which results synthesis can be played out as seen in Chapter 3. By using a spatial environment for the display and manipulation of the raw problem descriptions, results synthesis can take advantage of the highly developed spatial and visual abilities of most people (Marshall & Shipman, 1995) to express and perceive the evolving understanding of those who are performing the results synthesis. For many people, their first inclination when faced with the task of making sense of a large number of related elements is to attempt to arrange the elements spatially (Monty, 1990). When the entire dataset is displayed in the space, people can simultaneously consider all the data elements and easily switch between focusing on individual elements and considering larger groupings.

Requirement 17: Use spatial proximity and visual cues to express relationships amongst the data.

Spatial proximity is used to express relationships by arranging related things to be close together. That is, the distance between any two items on the space is a measure of how related the items are. Using spatial proximity to express relationships allows people to suggest a relationship by their relative placement. Thus tacit relationships can be expressed without the immediate need to explain or classify the nature of the relationship. There is no need to say why the two elements are related. The relationship is captured through their position in space, though it will eventually have to be made explicit if others are to understand what the relationship is. Visual cues are used to express relationships by making more important items more visually salient and by making similar elements have a similar appearance. In combination, these two mechanisms enable people to express a large number or relationships (Marshall & Rogers, 1992).

Requirement 18: Allow free form annotation of the underlying space.

Spatial expression by itself can be both powerful, and limited. Spatial proximity is limited in that it can only signify a single dimension of relatedness. In practice people want to express different relationships in different parts of the space. One area of the space may be for indicating duplicates, while another area is for indicating a general rule and instance of rule relationship. Spatial proximity is too powerful in that in some arrangements of elements in the space, the meaning of the relative positioning of elements is only meaningful at a threshold. For instance, overlapping may be used to represent some relationship, but the degree of overlap may not have any intended meaning. If the interpretation of the spatial relationships is not somehow controlled, then the degree of overlap is likely to be incorrectly interpreted as being significant.

There are two primary mechanisms for controlling the interpretation of layouts in a spatial environment. The first is through convention (Marshall & Rodgers, 1992). In tasks that involve the same people over a long period of time, convention may be an effective way of controlling interpretations of spatial relationships.

The second way of controlling the interpretation of spatial expression is through annotation of the space by allowing people to draw or write on the space. People use this ability to partition the space into different areas by drawing lines and/or providing labels. I believe that results synthesis is a relatively short-lived process, taking on the order of hours, and further that it will not involve the same group of people in every episode. Therefore, annotation is the best mechanism for controlling interpretation of the space.

Requirement 19: Allow the free creation and movement of data in the space.

When working with emergent ideas and concepts, people must face no overhead or impediment to their trying things out in the space. By allowing the free movement of elements in the space, people can express whatever relationships or structures they perceive in the data. Free movement allows people to express the current understanding of the relationships even though they may be tentative, ill-formed, or momentarily inexplicable. By removing barriers to adding elements to the space, people can add new elements that capture parts of the evolving understanding, such as generalizations or summaries of existing elements. Moran, Chiu, & van Melle (1997) have summarized this requirement as the design principle of "agility" for their work on supporting generic meetings of small groups. They argue that agility is important because it allows people to concentrate on expressing their ideas rather than on using the system or making their ideas fit the preconceived notions that have been implemented in the system.

4.5 Recording the outcome

The communication of the outcome of results synthesis can theoretically take place strictly through discussions and presentations, or in the opposite extreme, through a written document that is "thrown over the wall," that is, for which there is no possibility of subsequent discussion. The reality of most software development organizations is somewhere on a continuum between those two extremes, where a certain amount of the outcome will be recorded in writing and there will be limited discussions between the developers and the evaluators as necessary to further explain and understand what needs to

be done. As has been mentioned before, the amount of detail that needs to be recorded in the outcome will depend on many organizational factors, such as who is performing results synthesis (Section 4.6) and the existing relationship between the developers and usability. Strictly verbal presentations of the outcome severely limit the amount of detail that can be successfully communicated to the developers. Strictly written communication requires a great deal of work by those authoring it because of the great level of detail that must be included. Thus, a compromise is sought somewhere between the two extremes where each form of communication is used. The record of the outcome provides the basis for discussion, and the discussion expands and elaborates on the record, as needed. The requirements presented in this section aim to ensure that the known problems in communicating the outcome are avoided. As such, they provide a minimum level of what needs to be done, but are not being presented as comprehensive. More extensive requirements are beyond the scope of this thesis as they would require a more in-depth study of the many different development cultures and organizations.

Requirement 20: Document the outcome of results synthesis.

The outcome of results synthesis is ultimately the knowledge and understanding possessed by those who performed the results synthesis. However, it is important that this knowledge and understanding be documented in a concrete record at the appropriate level of detail. A tangible record of the outcome has many advantages. It can be widely distributed, repeatedly referred to, easily browsed and selectively studied. It also provides historical and educational reference, allowing others to see the evolution of the design, the issues that influenced the design, and how others have solved problems that may be similar to ones they are facing.

If the outcome of results synthesis is not documented and is communicated to the developers solely by verbal means, they will find it easier to treat the evaluation merely as a “rubber-stamp,” where they look for a “pass/fail” with the usability evaluators as gatekeepers in the development process (Sawyer, Flanders, & Wixon, 1996). If this happens, the developers may overlook, neglect, or ignore verbal recommendations that are made. By documenting the outcome, developers can be held accountable to a pub-

lic standard, and thus they are less likely to try to reduce all the recommendations to a single “yes” or “no” on the interface as a whole.

The process of preparing the documentation also benefits those performing results synthesis directly. It will help refine and clarify both the evaluators’ understanding of the interface and its shortcomings, and how they will communicate this understanding to development.

The documented outcome also fosters the relationship between usability and development by providing a basis for, and formal encouragement of, discussion of the problems and solutions raised in the outcome and the issues surrounding them. This will hopefully lead to greater appreciation of their respective concerns and expertise as well as the benefits of a closer working relationship.

The exact method used to document the outcome will depend on the organizational culture in which the Heuristic Evaluation is being done. In some cases a written report may be the best form of documentation (Sawyer, Flanders, & Wixon, 1996). In others, the outcome may best be recorded as entries in a problem tracking system (Muller, 1997).

Requirement 21: Provide separate accounts of problems and their solutions.

Getting development to consider a problem and effect its repair is aided greatly if the problem and the solution are described separately in the outcome. If the outcome contains only a set of problems, they may not be addressed because development does not have the conceptual or technical understanding to come up with a solution (Sawyer, Flanders, Wixon, 1996). If the outcome contains only a set of solutions, then the developers, who may not recognize the underlying problem, are put in the position of deciding whether the problem is repaired solely on the basis of whether the provided solution has an acceptable cost. This approach eliminates the opportunity to leverage the developers’ knowledge and understanding of the implementation, which might lead them to come up with an alternative, less costly solution if they were able to understand what the problem was. Hence providing only solutions will lead to fewer problems being fixed because their repair will be considered too expensive, when this

is not necessarily so. By separating the description of problems and solutions, both are likely to be expressed better, and the desired result is more likely to be obtained.

Often the raw problem descriptions are expressed in terms of a solution (Jeffries, 1994). By forcing the two concerns to be presented independently in the outcome, those performing results synthesis are encouraged to examined both issues more closely. This fosters the sorts of qualities called for in Section 4.3. With the description of the problem separated from that of its proposed solution, the problem may be more easily described in terms of users and tasks (Jeffries, 1994). This makes the justification of the problem more concrete. The separation also allows developers to consider alternate solutions and verify if they will also solve the problem (Jeffries, 1994). Often the solutions proposed by those performing results synthesis are non-optimal or infeasible because they do not have adequate knowledge of the constraints and assumptions driving the design of the interface. By allowing for alternate solutions, those who do have this knowledge – the developers – are allowed to use it to create solutions that address both their own concerns as well as those of usability.

Requirement 22: Include a description of the severity of each problem.

The reality of software engineering is that development does not have unlimited resources at its disposal to address all the problems in, demands upon, and concerns about the application being developed. The key task of software engineering is the careful allocation of developer effort to achieve the best possible product with the limited resources available. This implies trade-offs, which in turn imply prioritization. If a Heuristic Evaluation is to make a substantive contribution it must be easy for development to understand the seriousness of each problem raised in the outcome and the consequences of not repairing it. Thus the outcome usually includes an indication of problem severities.

At a minimum, each problem report in the outcome includes a numeric rating of severity (Nielsen, 1994). Problems reported by engineering quality assurance activities also have numeric severities, and consequently using numeric severity ratings for

usability problems can give the outcome a more familiar and professional appearance (Sawyer, Flanders, & Wixon, 1996).

The danger is that by reducing the communication of severity to a single number, developers are unable to check whether they agree with the assessment of the severity as they do not have access to the justification used by the evaluators. If those preforming results synthesis and the developers of the interface being evaluated do not have the same understanding of the meaning of the rating levels, the developers can downgrade or dismiss the problems reported by usability.

For this reason, Jeffries (1994) recommends that the severities be more verbose descriptions in terms of users and tasks. This allows the developers to set priorities better and to understand the likely consequences of their decisions about whether to fix the problem or not. She recommends the extended exposition for severities because numeric severities provided by individual evaluators are often highly inconsistent due to the differing biases and assumptions of the evaluators.

Numeric severities may be used in the outcome of results synthesis, provided that those performing results synthesis have good reason to believe that the ratings given will be viewed as accurate by the developers. One way of having a good reason is to include a developer amongst those performing results synthesis (Requirement 25).

In situations where there is a risk of divergence between the usage of numeric severities in the outcome and their interpretation by developers, I recommend the more verbose description of severities be used. In discussing Requirement 9, I argued that usability engineering evaluations and more traditional engineering quality assurance activities are fundamentally different. It is because of this difference, and the fact that usability engineering in not a long established activity in most development organizations that the use of numeric ratings in the outcome of results synthesis can be problematic, even though it is accepted practice for engineering QA activities.

4.6 Who performs results synthesis

The requirements stated in the preceding sections do not place any restrictions on the number people involved in results synthesis. Nor do they say anything of about who

should be involved in it. Clearly the choice of an individual versus a group analysis, and the choice of group composition will have an effect on the process.

Nielsen (1994) does not have the same concept of results synthesis as presented in this chapter. He does, however, discuss a number of the subprocesses that correspond somewhat to results synthesis. He suggests that multidisciplinary "debriefing sessions" may be held after the inspection phase to generate solutions to the problems found in the inspection. In order to obtain severity ratings, he suggests that a single individual "synthesize" a unified list of problems found in the inspection phase. This list is then resubmitted to the inspectors for their rating.

Having a single individual perform results synthesis is, I believe, the most common situation in industry today for several reasons. First, it seems more efficient. There is no need to schedule a meeting, round up the participants, or deal with all those potentially troublesome group interaction issues. Secondly, the end result may also be more coherent as it is the result of a single person's opinions, ideas, vision, and writing style.

However, having an individual perform the results synthesis is not without its problems. That single person must read and comprehend all the raw problem descriptions. This may not be a particularly easy task (Jeffries, 1994), as the raw problem descriptions may be too vague for the individual to understand. These problems will either have to be discarded from consideration, or additional effort will have to be made to decipher them. Having one of the inspectors perform results synthesis may speed the process up, since that person will be familiar with their own subset of the raw problem descriptions. However, this may bias the process as the inspector is likely to have preconceived ideas of the problems and their solutions as a result of inspecting the interface. If this bias is eliminated by having results synthesis performed by someone other than the inspectors, then there is increased time spent reviewing the inspectors' raw problem descriptions (Jeffries,1994).

Given that many practitioners do not even muster the minimum suggested number of evaluators (Nielsen, 1995a), is it realistic to suggest that results synthesis be carried out by more than one person? I believe that the answer to this question is "yes." I believe this to be true because adding more evaluators at best leads to an incremental improvement in the

number of problems found (Nielsen, 1994b). This does not necessarily translate into an improved product. Results synthesis, performed by a group, will, I believe, lead directly to an improved product.

Requirement 23: Results synthesis is participatory.

There are many different stakeholders interested in the outcome of the Heuristic Evaluation, and as many as possible should be included in results synthesis. This is in line with the accepted approach to participatory software development (Schuler and Namioka, 1993). The evaluators and the developers are certainly stakeholders, and their participation is essential to encourage "mutual understanding." Additionally, technical writers, customer support engineers, sales & marketing, and of course the end users can be viewed as stakeholders. The philosophy of participatory practice is that each stakeholder has something to contribute. The literature on Heuristic Evaluation has recognized the value of contributions by two particular stakeholder groups in addition to usability specialists: developers (Nielsen, 1994b) and end-users (Muller et al, 1998).

Requirement 24: Include the inspectors in results synthesis.

The inspectors of the interface who generated the raw problem descriptions should certainly be included in results synthesis. They are the primary interpreters of the raw problem descriptions. They also are usability specialists (Nielsen, 1992), and can represent the usability point-of-view in results synthesis. As usability specialists, they are the primary translators of user needs into development direction (Muller, 1998).

Requirement 25: Include developers in results synthesis.

Developers have an immediate and obvious contribution because they are the most knowledgeable about the existing design and implementation and they are the ones who will have to make any changes. Many of the shortcomings in problem reports identified in the preceding sections can be ameliorated by having developers involved in results synthesis. Since developers are not usually involved in inspections (Jeffries, 1994) they can be profitably engaged in results synthesis.

The goal of results synthesis is effective communication of the usability perspective on the interface—its problems, their possible solutions, and the urgency with which

they are to be addressed—to developers. By having developers participate in results synthesis both the quantity and quality of the communication is increased. Including developers in results synthesis increases the quantity of communication by having developers and the usability specialists engaged in the wide ranging discussions about all aspects of the interface that are necessary to performing results synthesis. This communication is in addition to the formal presentations of the outcome of results synthesis to the developers as well as the informal discussions that follow. Developers who are aware of the benefits of usability evaluations seek more interaction with usability engineers (Sawyer, Flanders, and Wixon, 1996), and participatory results synthesis is an opportunity to fulfill this desire.

Involving developers in results synthesis improves the communication of the outcome to development by helping to ensure that the needs and concerns of developers are met by the outcome. In creating the outcome, representatives of development are present to ensure that the outcome meets the needs of developers and addresses their concerns. This is analogous to the way that including users in participatory design aids in producing software that is appropriate and acceptable to the end users.

Involving developers in results synthesis not only improves communication, but increases the likelihood that development will act appropriately on the basis of the outcome of results synthesis. Developers will take more ownership of the result (Dumas & Redish, 1993). Developers will have increased confidence in the result because they had a say in its creation. Developers will see that results synthesis includes careful, informed consideration of the problems and solutions, leading to further confidence in the result and hence an increased probability that they will take the appropriate action.

A caveat to including developers is that good things will not come from haphazard inclusion of any development representative. The person or persons selected will depend on the particular context. In general the developer or developers should be someone who is respected by the other developers, and has appropriate regard for usability activities.

Requirement 26: Include end users in results synthesis.

Involving end-users in results synthesis is a more daring proposition, though it represents the true spirit of participatory design. For results synthesis, the primary benefit of including end-users is that the descriptions of problems and severities in terms of users and tasks have increased veracity and cache. The end-users are also able to forestall debates about what users are really like and what they really do that may otherwise side-track or derail the process.

Requirement 27: Systems supporting results synthesis are groupware enabled.

In Section 2.4, I suggested that one barrier to gathering a sufficient number of inspectors in practice was that qualified inspectors may be spread throughout many different geographic locations. By allowing the participants of results synthesis to remain at dispersed geographic locations, groupware systems for results synthesis enable a participatory approach to the process (Requirement 23) without the substantial costs to the organization of bringing all the participants to one location. Thus, systems supporting results synthesis as it is described in this chapter must be groupware enabled to offset the cost of adopting a participatory process. To increase the acceptance of a participatory results synthesis process, its costs should be minimized as usability practitioners are very conscious of return-on-effort (Sawyer, Flanders, & Wixon, 1996).

Systems supporting results synthesis that are not groupware enabled will encourage the division of labour that is warned against in the discussion of Requirement 5. Systems that are groupware enabled will allow the participants to exploit the advantages over traditional media that system support provides while still allowing the participants to interact with each other and the artifacts in their workspace.

Groupware is often analyzed on the asynchronous versus synchronous collaboration continuum. Asynchronous collaboration takes place when the users of the system view and manipulate the work artifacts at different times, while synchronous collaboration occurs when the users of the system view and manipulate the same collection of work artifacts at the same time. The success of participatory endeavours such as results synthesis depends on the ability of the participants to translate their ideas and issues into

each others language (Saarinen & Saakjarvi, 1989). This can only occur in an environment that allows efficient grounding of communication (Clark, 1997). By allowing simultaneous and transparent reference to common artifacts, synchronous groupware can enable efficient grounding of discussion about the work artifacts. Therefore, in this thesis, I will concentrate on synchronous groupware support for the results synthesis process.

4.7 Conclusion

In this chapter, I have presented what I believe to be the essential requirements for an effective Heuristic Evaluation results synthesis process, and system support for that process. These requirements are summarized in Table 4.1 on page 77.

The requirements were grouped into five categories. The necessary elements of raw problem descriptions which form the basis for results synthesis were presented in Section 4.2, as the first group. The second group of requirements (Section 4.3) enforces the elimination of undesirable properties that are present in the mass of raw problem descriptions. The desirable qualities that meeting these requirement produces were summarized as completeness, coherence, and conciseness. Obtaining these qualities is not a simple process, and they may emerge only after extensive processing of the raw problem descriptions. Enabling the emergence of these qualities is the purpose of the third group of requirements (Section 4.4). Supporting emergence is best done through using space as the primary organizational tool where spatial proximity and visual cues are the primary means of expressing relationships, the space can be annotated, and elements freely rearranged and created. Results synthesis is not complete without the production of final problems reports, and it is the form in which these reports are recorded that is the subject of the fourth group of requirements (Section 4.5). The final problem reports must be documented, must separate the description of the each problem from the description of its solution, and must describe the severity of the each problem. The question of who is to participate in results synthesis is addressed by the last group of requirements (Section 4.6). A participatory approach to results synthesis where the inspectors of the interface as well as its developers and end users are included in the process is recommended.

In Chapter 3, I presented a scenario describing results synthesis in a paper-based environment. The environment and the materials shown in the scenario meet the requirements set out above. In Chapter 5 I will describe a prototype groupware results synthesis system designed to create a computational environment that satisfies these requirements.

Elements of raw problem descriptions
Requirement 1: Include a description of the problem.
Requirement 2: Include the associated heuristic.
Requirement 3: Provide the inspectors with a space to record possible solutions.
Requirement 4: Include an identification of the author.
Ensuring a quality outcome
Completeness
Requirement 5: Consider the entire collection of raw problem descriptions.
Requirement 6: Consider all the operative constraints on the interface.
Requirement 8: Consider all the applicable trade-offs in the interface.
Requirement 7: Provide adequate justification for every problem.
Requirement 9: Generate workable solutions to the problems raised.
Requirement 10: Generate alternate solutions for problems where possible.
Coherence
Requirement 11: Create a consistent set of underlying assumptions.
Requirement 12: Create consistent language and terminology.
Conciseness
Requirement 13: Eliminate duplicate problem reports.
Requirement 14: Express problems and their solutions at the right level of abstraction.
Requirement 15: Examine ambiguous problems.
Enabling emergence in results synthesis
Requirement 16: Provide a spatial environment.

Table 4.1: Summary of requirements

Requirement 17: Use spatial proximity and visual cues to express relationships amongst the data.
Requirement 18: Allow free form annotation of the underlying space.
Requirement 19: Allow the free creation and movement of data in the space.
Recording the outcome
Requirement 20: Document the outcome of results synthesis.
Requirement 21: Provide separate accounts of problems and their solutions.
Requirement 22: Include a description of the severity of each problem.
Who performs results synthesis
Requirement 23: Results synthesis is participatory.
Requirement 24: Include the inspectors in results synthesis.
Requirement 25: Include developers in results synthesis.
Requirement 26: Include end users in results synthesis.
Requirement 27: Systems supporting results synthesis are groupware enabled.

Table 4.1: Summary of requirements

Chapter 5: Design

5.1 Introduction

This chapter will discuss the key design points in the **P**rototype **Re**sults **S**ynthesis **S**upport (PReSS) system. In Chapter 4, I put forth a number of requirements on the results synthesis process and any system that seeks to support it. In this chapter I will describe the main design points of PReSS and how they relate to the requirements I put forth in the preceding chapter. My discussion will be structured around the features of the system as they come into play during a scenario of use. I will follow the scenario I described in Chapter 3, showing how the system is expected to be used in similar circumstances.

Before presenting the system in the scenario of use, I will describe the system in general terms (Section 5.2). In Section 5.3 I describe how participants prepare for results synthesis when using PReSS. In this scenario, they use another system to capture the raw problem descriptions. The familiarizing step is described in Section 5.4. PReSS automatically primes the workspace with the problem descriptions, leaving the users only to review the entire collection. Section 5.5 describes the system support for the emergence stage of results synthesis. How the finalizing stage is supported is covered in Section 5.6.

The data set shown in this chapter was created by the author transcribing the set of raw problem descriptions created by the inspection process for the first study of results synthesis in a paper-based environment reported in Section 3.6. The set of heuristics used in that study were different than the set used in the capture system, so the author had to also perform translation between the two sets of heuristics in coming up with data set.

5.2 System overview

In this section I will provide an overview of the major design decisions in the system and their motivation. The environment the system provides is intended to be primarily spatial (requirement R16). It is based on the metaphor of the paper environment used in the studies and scenario described in Chapter 3. I have endeavoured to keep the system as simple as possible for two reasons: to meet my requirement (R26) of not excluding infrequent computer users through a complex interface and because I believe that simplicity leads

more quickly to skilled performance as simple tools are more likely to be ready-to-hand (Winograd and Flores, 1986). I have also tried to exploit the added power of computer representation where possible.

In keeping with the metaphor of the paper-based environment, problem descriptions are represented in the workspace by *cards* (See Figure 5.1), which model the index cards used in the paper scenarios. The users can freely create textual and free-hand annotations, and can also move most elements about workspace without constraint – as they can in the paper-based environment. One extension of the metaphor is the problem report element, which can be created in the workspace to record final problem reports at the close of the process.

Most of the functionality of the system is present in the workspace, and is accessed through direct manipulation or pop-up menus. The menubar contains all the system defaults provided by GroupKit (Roseman and Greenberg, 1996) as well as four commands specific to PReSS that are added to the **File** menu. These commands are generally used only at the very beginning and end of the process to move data into and out of the system respectively:

Open: Open a preexisting data file that contains a previously saved version of the workspace.

Save: Save the state of the current workspace in a file with a provided name.

Import: Bring problem description data into the system from a data file created by the capture system.

Publish: Create a set of HTML pages containing the contents of final problem reports in the current workspace.

During the process, the users concentrate on the three different views of the workspace that the system provides:

- **The main view**. This is the largest single component of the display as it is the most important (Figure 5.1, left side). The main view shows a subspace of the entire workspace. It is where annotations and problem reports are created, and content is added and deleted. The workspace has a fixed size, with the main view showing somewhere

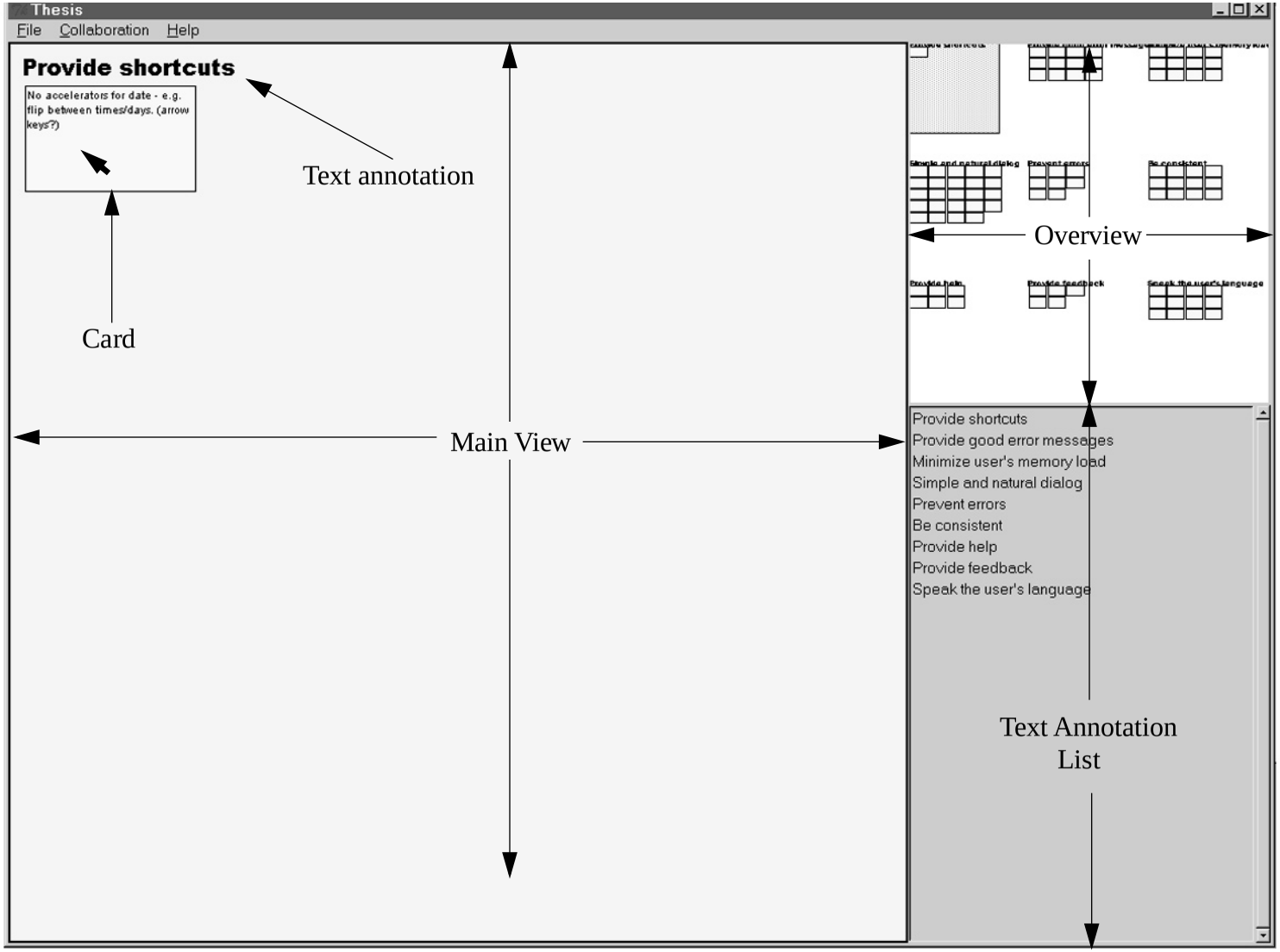


Figure 5.1: PReSS with sample data set loaded

between 1/16th and 1/11th of the area of the whole workspace, depending on the version of the system as well as the on which operating environment it runs. I had originally intended to provide a workspace of unlimited size. However, due to technical, design, and resource limitations, I decided to adopt a fixed size workspace for the proof-of-concept system. I chose an area large enough to give plenty of room to handle the size of data sets I expected to be used in the evaluations, based on my observations of paper based results synthesis.

- **The overview**. The entire workspace is presented in the overview (Top right, Figure 5.1). The intent of the overview is to provide workspace awareness (Gutwin, 1997), both for the individual to provide context for their actions and access to parts of the workspace not in the main view, as well as for staying aware of and for interacting with collaborators. The entire contents of the workspace are scaled to fit into the overview. This scaling is proportional, except for the text annotations. These appear much larger in the overview than they actually are in the workspace. This is done because the true scaling factor is so great that they would be reduced to black bars if accurately scaled. By exaggerating their size in the overview, I believed that users would still be able to recognize the annotation even if they were not readable, due to redundancy in written English (Wickens, 1992).
- **The text annotation (TA) list**. This listbox in the lower right-hand corner of Figure 5.1 contains a list of all the text annotations in the workspace. It is designed to provide navigational and organizational shortcuts. I will discuss the motivation and function of the TA list in more detail later.

In creating the system I tried to keep it as simple as possible. I aimed to create a system that relied on a few mechanisms that users could exploit in many ways to achieve the richness of interaction and expression they would need to carry out results synthesis. This simplicity would make the interface easier for everyone to use, thus removing barriers to participation in the process (R23), as I expect that the system will not be frequently used by any individual. The simplicity is also aimed at allowing those end users with less experience and confidence in using computers to participant fully in the process (R26). PReSS

is a groupware enabled system (R27). The envisioned scenario of its use is that it is often hard to gather the desired group of experts in one place at one time to perform results synthesis. By using PReSS, the various experts and representatives one ought to include in results synthesis can participate even when they cannot meet face-to-face.

5.3 Preparation

The preparation stage of results synthesis is concerned primarily with ensuring that the raw problem descriptions on which the process operates have the necessary properties. The requirements that deal with this stage are summarized in Table 5.1.

Elements of Raw Problem Descriptions
R1: Include a description of the problem.
R2: Include the associated heuristic.
R3: Provide the inspectors with a space to record possible solutions.
R4: Include an identification of the author.

Table 5.1: Requirements affecting preparation

PReSS deals with preparation by getting its input from another system which was created by Saul Greenberg, my supervisor and an interested party in the research. By using a capture system (see Figure 5.2) and by being able to import its output without further processing, PReSS aims to remove any disincentive to inspectors in participating in results synthesis (R24).

The problem description capture system is shown with the data set that will be used throughout the chapter. This data set is the one produced by the inspection process of the group that was part of the first observational study reported in Section 3.6. This data set is a super-set of the data shown in the scenario in Chapter 3. This data set is also used in a number of the studies discussed in Chapter 7. A detailed discussion of the design of this system is not warranted because it is a simple system of conventional design whose sole purpose is to allow Heuristic Evaluation inspectors to log the problems they find in an evaluation. I mention the system because it is the only currently supported way of creating

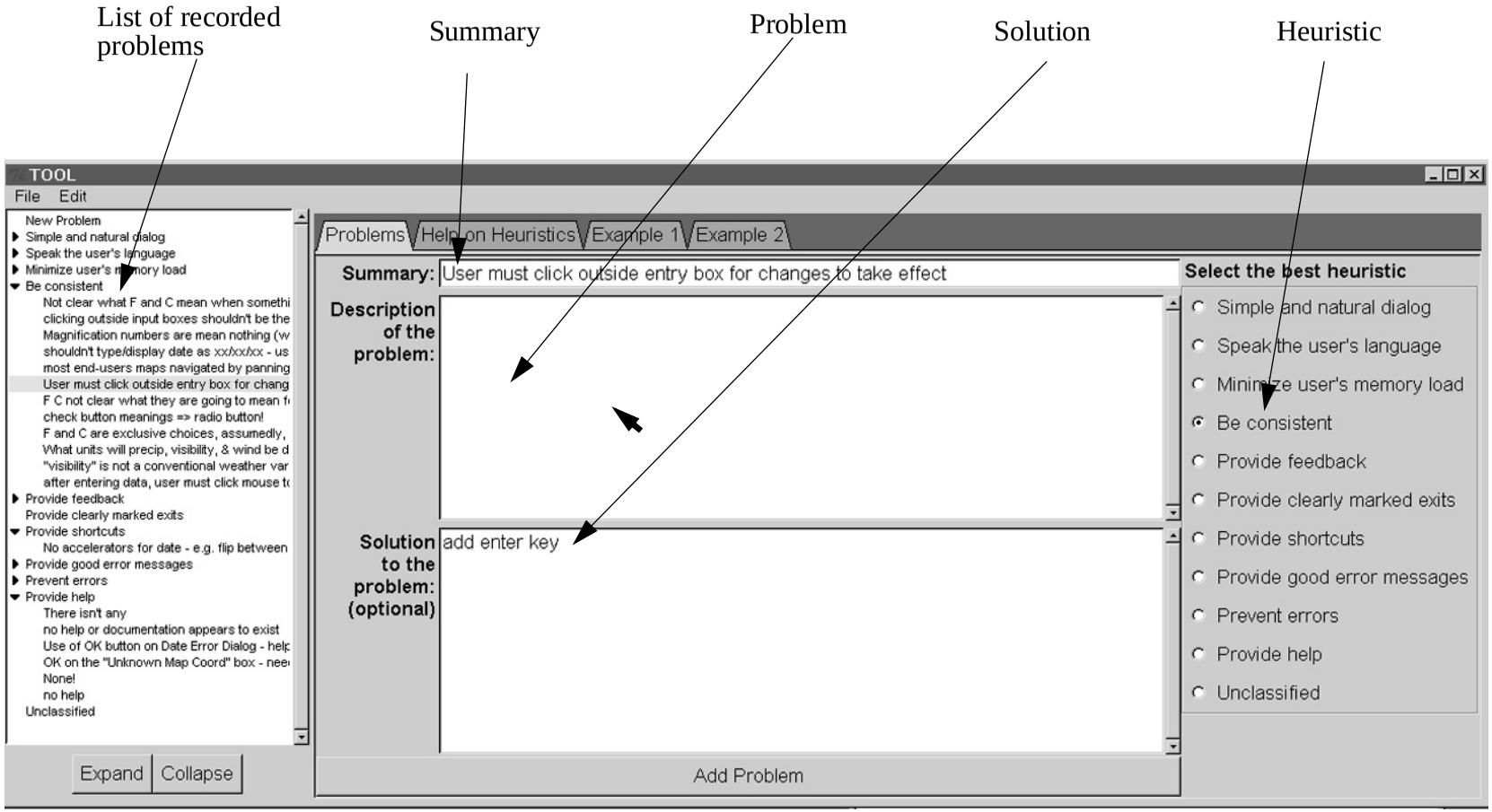


Figure 5.2: Capture system with data set loaded

data that can be read by PReSS. What is important about this system is that its design has been influenced so that its output conforms to the requirements of results synthesis listed in Table 5.1. Note that the capture system requires at least a summary description of the problem to be entered (R1) and provides space for a more verbose description as well as a separate space for notes on solutions to the problem (R3). Each problem reported must also have an associated heuristic (R2), indicated by selecting one of the radio buttons on the right. All of this data is preserved when it is saved out of the capture system and then imported into PReSS. However, the capture system does not provide identification of the author of problem descriptions, and as a consequence neither does PReSS (R4).

The capture system records four pieces of information in each problem description (Figure 5.2) and I evaluated each of these to see whether or not I needed to display it on the card:

- **Summary**. The capture system requires that the summary field be filled in for every problem description recorded. Most of the problem descriptions recorded in my observational studies of results synthesis in paper-based environments were very short and would easily fit into the summary field. I wanted to have the cards be of fixed and relatively small size for technical and design reasons. Since the summary field would limit the amount of content entered, it would mesh nicely with the above stated goals and therefore I chose to display this field on the cards.
- **Problem**. The problem field in the capture tool was seldom needed as the problem descriptions seen in the observational studies from Section 3.6 were almost always small enough to fit into the summary field. For technical and design reasons I did not want to have a large amount of text displayed on the card, so I decided that if there was in fact content in the problem field, not displaying it would not have an adverse impact on the results synthesis process. If the participants felt that they did not understand the summary description of the problem, then they could open the edit window (Figure 5.3) to see if further explanation of the problem was available.
- **Solution**. The solutions that are recorded do not play a role in the emergence stage, but are needed only in the creation of the final problem reports. In my studies of paper

based results synthesis, the noting of solutions separate from problem descriptions was infrequent. In fact, the participants often recorded solutions instead of descriptions of the problem (Jeffries, 1994). Thus I felt the solution field did not need to be shown on the card, but should still be retained for reference at the end of the process.

- **Heuristic**. In the paper-based environments, the main role of the heuristics was to organize the initial layout. The participants in these studies told me that they did not pay attention to the heuristics after making the initial layout. Thus I concluded that it was not necessary to display the heuristics on the card. The heuristics are retained, however because they may help readers make sense of the problem description, and may be used in creating the final problem reports. Their utility is confined to the ends of the process and is not apparent in the emergence stage which is what the cards are primarily aimed at supporting. The information is already present in the initial layout of cards in the workspace where all the cards reported with a given heuristic are grouped in the workspace under that heuristic as a text annotation as seen with “Be Consistent” in Figure 5.3.

Figure 5.3 shows PReSS after importing the data shown in Figure 5.2 from the capture system output. The cards, which are the means by which raw problem descriptions are represented in the workspace, show the summary description. The rest of the problem description content can be viewed by double-clicking on an card or choosing “Edit” from the card’s pop-up menu (Figure 5.3). This action creates a window that displays the entire content of the problem description. My guiding principles in designing the system was that I wanted to be able to fit as many cards as possible in the main view while still having the content within it legible and readable, and that different cards be easily distinguishable as they are in the paper based environment. My studies of results synthesis in paper based environments lead me to believe that “hiding” some of content in the problem description is a net gain. I wanted to keep the amount of information displayed on the card to the minimum necessary.

By minimizing the content displayed on the card I was able to maximize my trade-off between number of cards in the main view and the legibility and readability of the text on

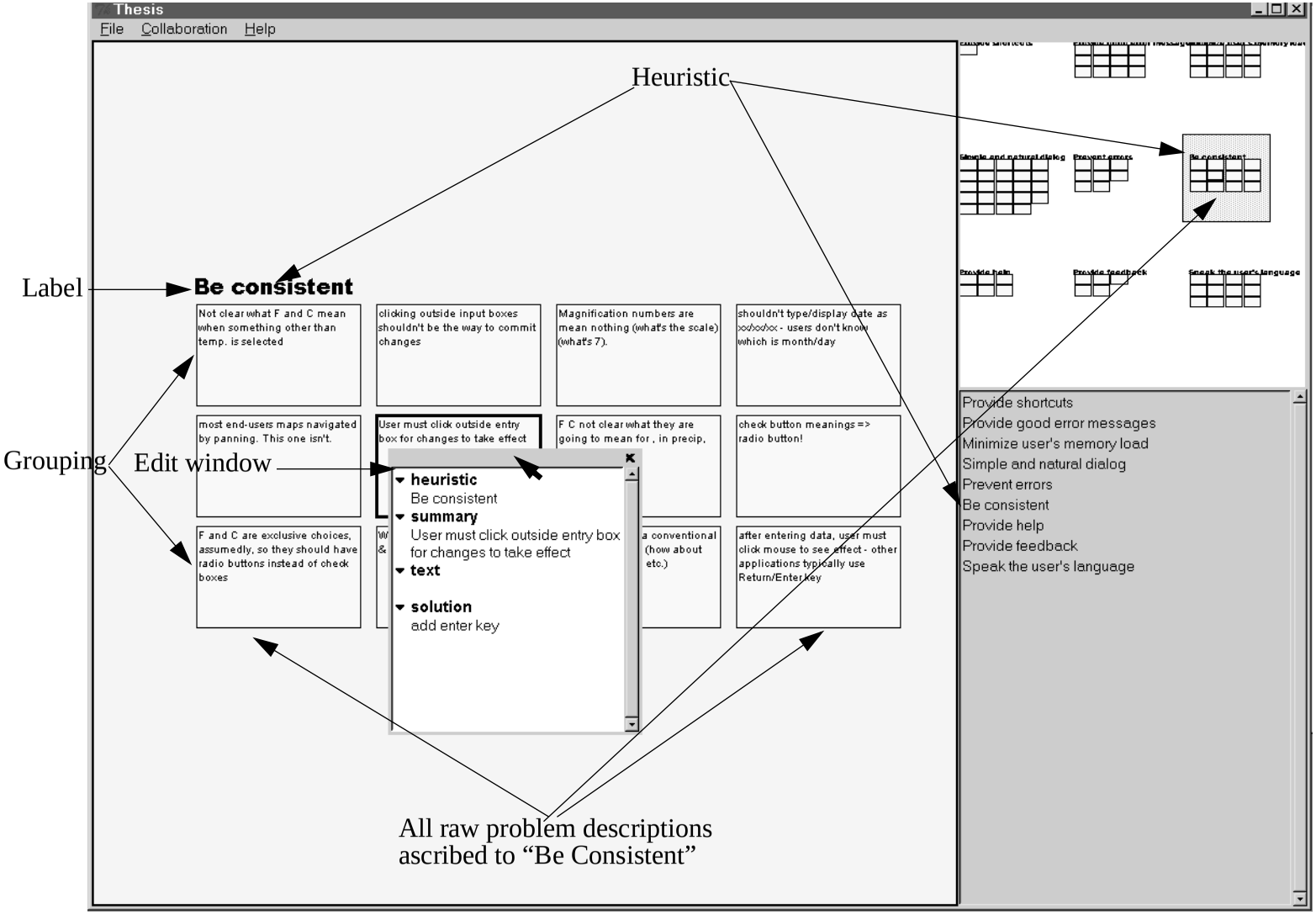


Figure 5.3: Cards laid out by heuristic with edit window for one card displayed

the cards. Since the cards were of fixed size, the less content shown, the larger the type size that could be used. This decision is one in a series that involve the fundamental trade-off in the design of PReSS. This trade-off is between legibly showing detailed information and showing as large an area as possible, and hence as many data points as possible. In general I have tended to favour the latter, believing that emergence is best supported by seeing as many different data points as possible, and that spatial environments are more effective when they show as much of the underlying space as possible.

5.4 Familiarizing

As described in Chapter 3, there are two steps to the familiarizing stage in paper based results synthesis: placing the problem descriptions on the work surface and then reviewing the entire collection.

The first step is done automatically by the system whenever a set of raw problem descriptions is imported. PReSS creates a layout of the raw problem descriptions, grouping them according to heuristic (Figure 5.3). PReSS also creates text annotations for each heuristic grouping. This is done to provide landmarks for navigating about the space and to aid the users in making references to locations in the workspace. Also, the text annotations can be deleted, so they are less permanent and disruptive than if they were made in a paper based environment. I expect as the emergence stage unfolds that users will create their own text annotations to label emerging groupings and will eventually dispense with the heuristic annotations.

With the entire collection of raw problem descriptions primed in the workspace, the next step is for the users, who are synonymous with the participants in results synthesis, to review the entire collection (Figure 5.4). The familiarizing stage is done in straight analogy to how it happens in paper based environments – the participants move their focus of attention around the workspace until they have considered all the data (R5). In the system, the user's focus of attention is synonymous with the location of their main view. The main view can be repositioned using one of three mechanisms:

1. *Panning the main view.* The position of the main view in the workspace can be changed by dragging in the main view with the middle mouse button. This follows a

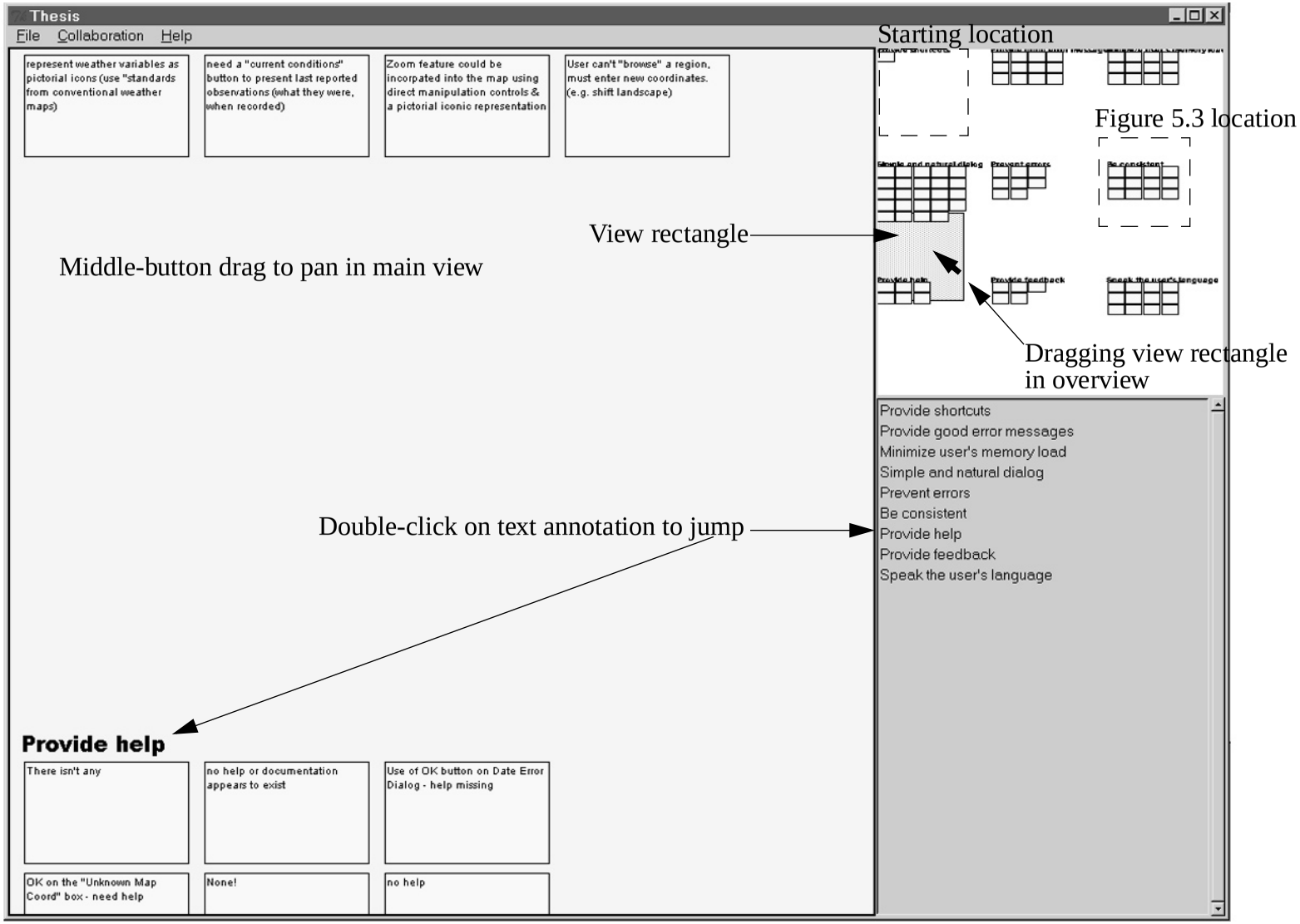


Figure 5.4: Navigating the main view

"drag the document" model, where moving the mouse to the right moves the view to left. The analogy is that by moving the mouse you are moving the document under a fixed viewport. This is quite common among certain applications, especially on the Macintosh.

2. *Dragging the view rectangle*. Within the overview, a view rectangle indicates where in the entire workspace the main view is currently located. For example, in Figure 5.4 the user have moved the location of the main view from the "Be Consistent" grouping (Figure 5.3) to in-between the "Simple and natural dialog" and the "Provide Help" grouping by dragging the view rectangle with the left mouse button. The overview enables the user to establish the spatial relationships between the groupings, in this case that "Be Consistent" is in the middle-right, and that "Provide Help" is in the lower-left.
3. *Jumping using the text annotation list*. In the lower right corner of the system interface is a list of all the text annotations in the workspace. This list is an organizational and navigation shortcut. The organizational aspect will be covered in the next section. The entries in the list function as navigational shortcuts when the user double-clicks on an entry the main view is automatically moved so that the text annotation is visible in the main view. For example, if the user wanted to see the area around a text annotation "Provide Help," double clicking on the "Provide Help" entry in the text annotation list would move the main view so that the annotation was within the view (Figure 5.4).

In the course of the familiarizing stage, the participants may come across duplicate problem reports. In this case, the user has located a number of duplicate raw problem descriptions noting that there is no online help provided by the system being evaluated. The cards are selected by Control-clicking on them with the left mouse button (Figure 5.5). If the target collection of cards had been on their own without other intervening cards, the user could have used Shift-drag to select the contents of the area the user dragged out. These are standard Microsoft Windows™ conventions. The user then posts the pop-up menu (Figure 5.5) using the right mouse button, in order to put them into a pile (Figure 5.6). When this happens, the best representative is not on top of the pile, so the user posts the

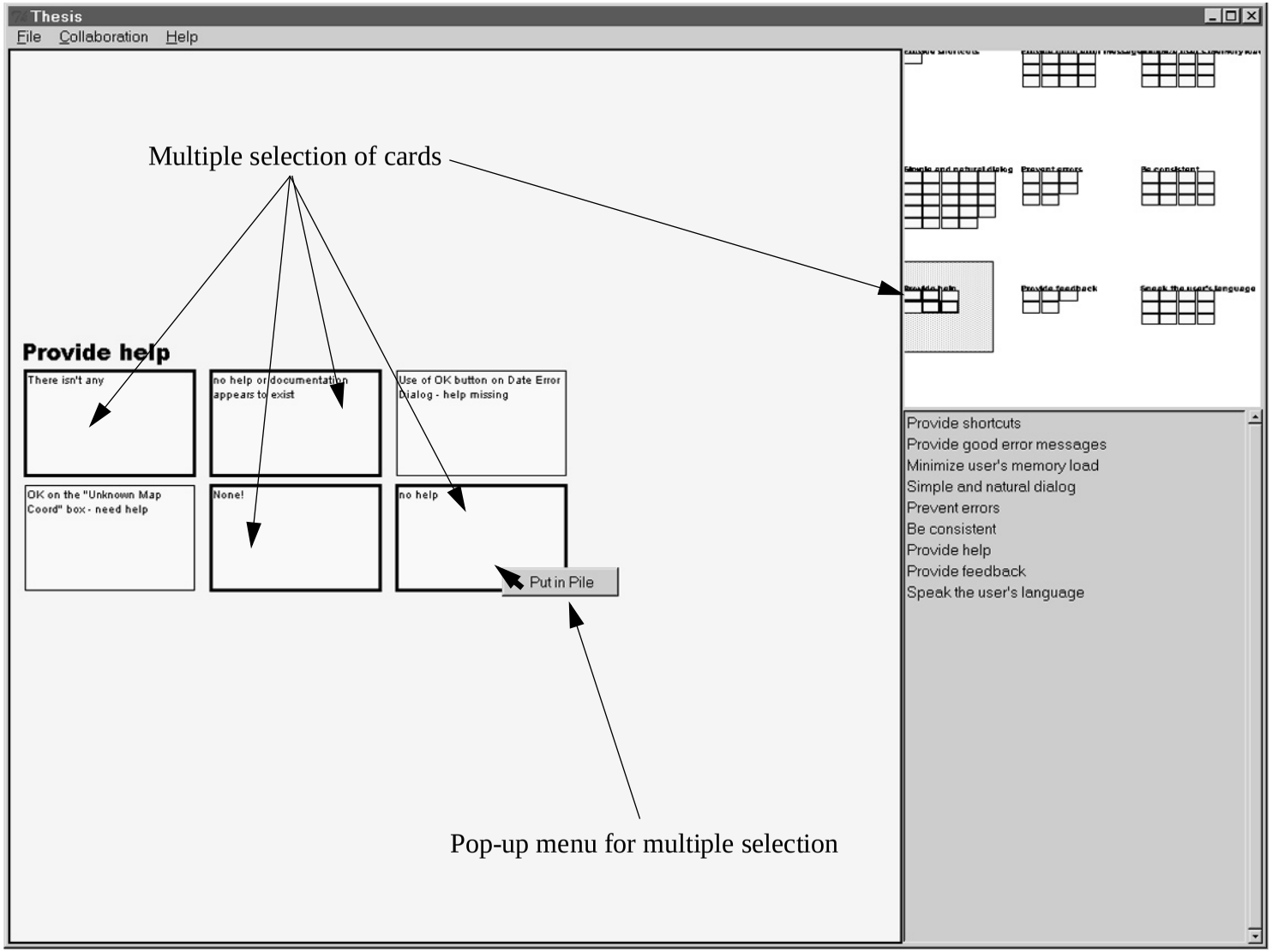


Figure 5.5: Selecting multiple cards

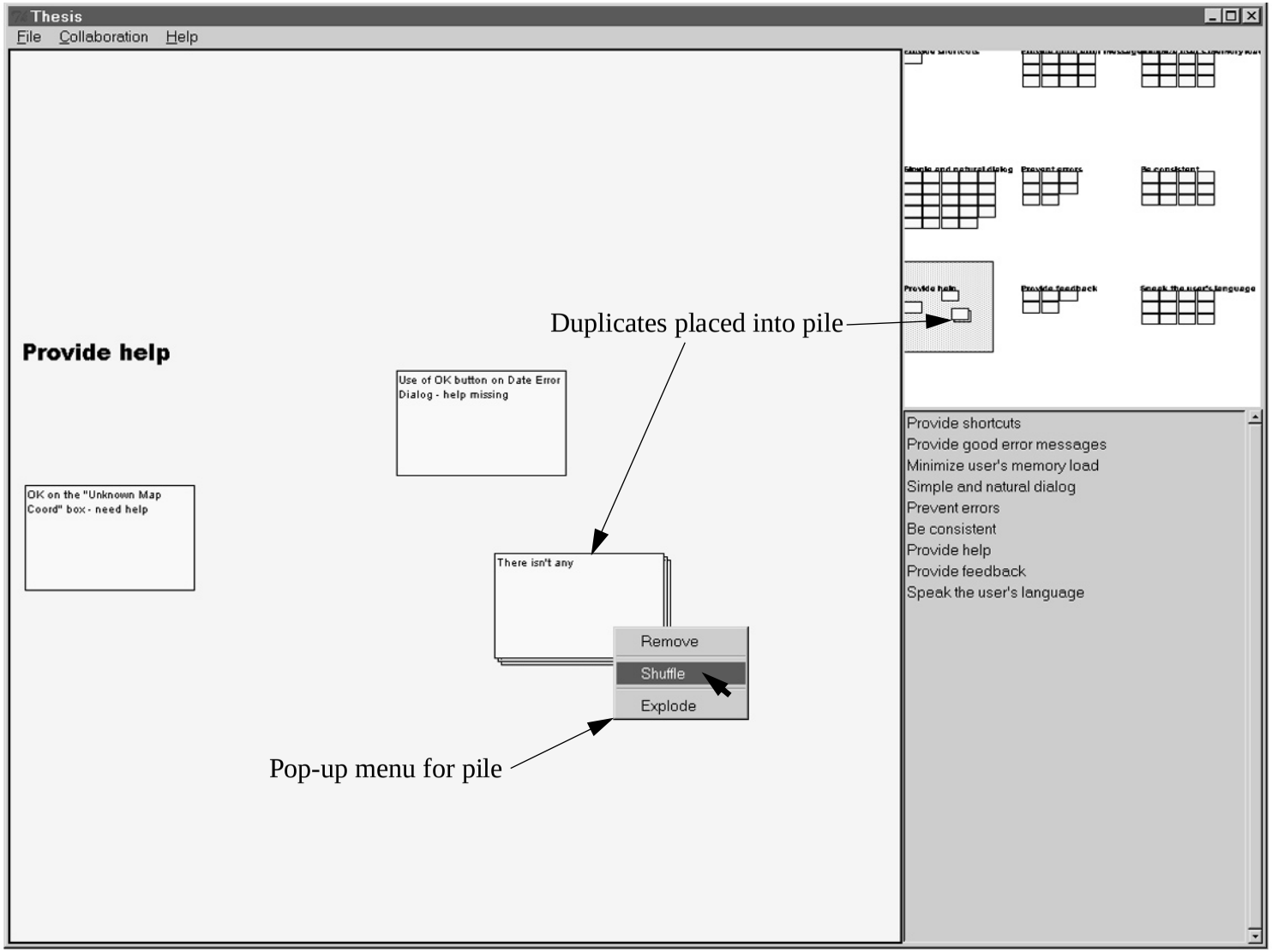


Figure 5.6: Pile created, menu posted

pop-up menu for the pile and chooses *Shuffle* until the desired representative is on the top of the pile (Figure 5.6). Thus duplicate raw problem descriptions can be eliminated (R13) by placing them into a single pile. Once the pile has been created, additional cards may be added to a pile by dropping them on top of the pile. Piles can be freely moved throughout the workspace. In addition to *Shuffle*, the pop-up menu on piles (Figure 5.6) provides two other actions:

1. *Remove*: takes the top card off the pile and places it in the workspace as an individual card.
2. *Explode*: separates all cards in the pile into a selected group of individual cards.

After all the easy duplicates have been identified and put into piles (Figure 5.7, compare to Figure 3.6), and all of the participants have reviewed the entire collection of raw problem descriptions, the familiarizing stage is complete and the participants move on to the emergence stage.

5.5 Emergence

PReSS is not the first system to seek to support emergence in group activities. The design of PReSS is deeply influenced by the spatial hypertext system VIKI (Marshall, Shipman, and Coombs, 1994; Marshall and Shipman, 1995). VIKI employs visual aliases in a spatial workspace as the primary mechanism of representing data and its meaning. However, VIKI is focused more on longer-term loosely-coupled group work and does not provide support for real-time synchronous work, unlike PReSS. Tivoli (Moran, Chiu, van Melle, 1997) is a system designed to run on an electronic whiteboard. The system is intended to support groups in real-time co-located intellectual activities that feature emergence. PReSS, on the other hand, is designed to support distributed groups, a situation Tivoli is ill-suited to handle as it lacks important awareness support features (Gutwin, 1997). GUNGEN (Munemori and Kagasawa, 1996) is a system design for real-time collaboration amongst distributed groups performing the KJ method, a technique for organizing ill-structured data. The KJ method is similar to results synthesis, though aimed at different, though related, kinds of activity. PReSS provides much better support for collaboration over the workspace (Gutwin, 1997) that is lacking in GUNGEN. While there are a number

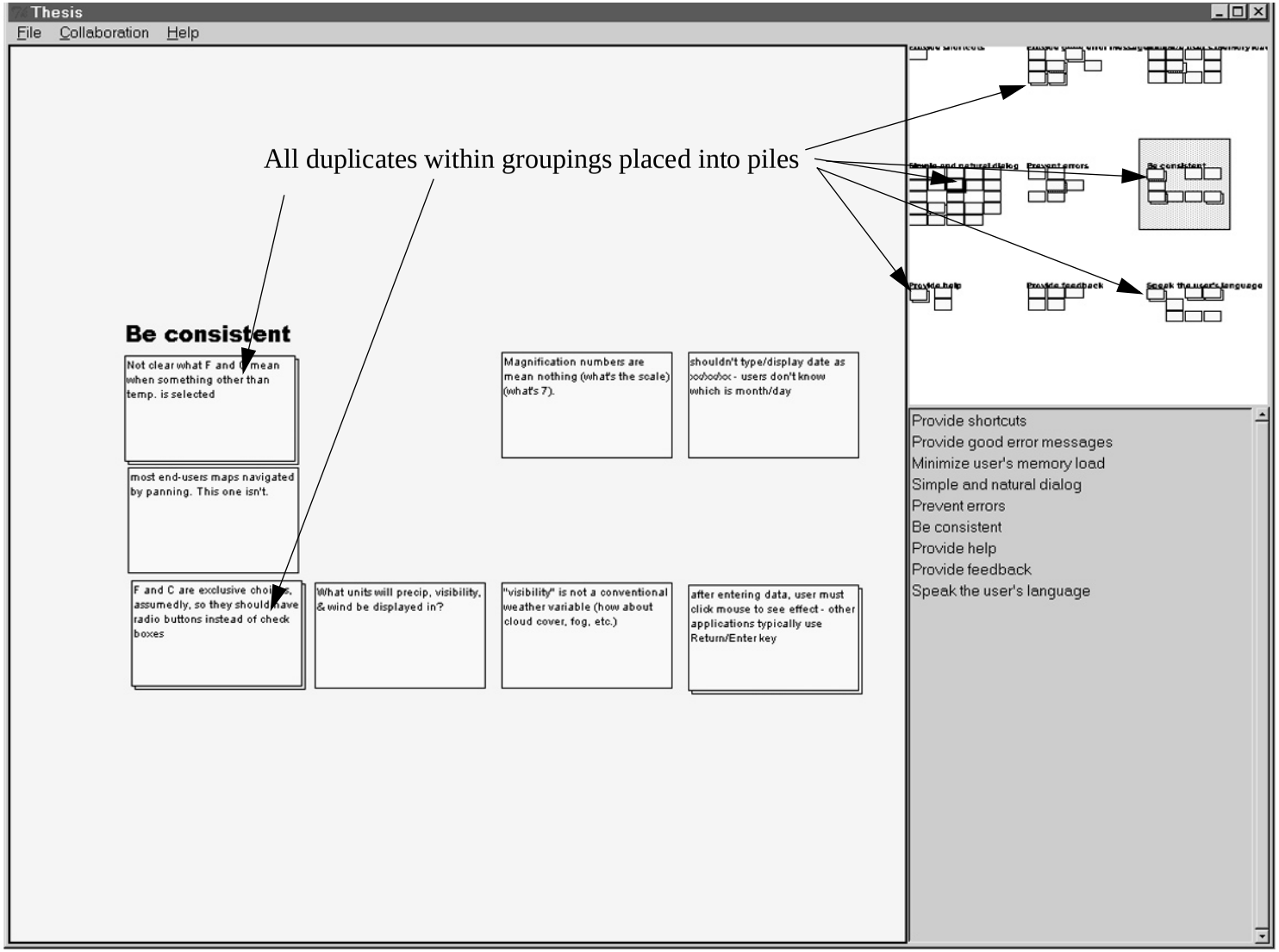


Figure 5.7: Familiarizing complete, duplicates put into piles

of general systems that support one aspect or another of results synthesis, none addresses the situation for which PReSS is designed. PReSS is also designed to exploit the properties of the raw data from Heuristic Evaluation to enhance support for results synthesis, something that more general systems, such as those discussed above, cannot do.

Returning to the scenario of use, the first thing these particular users do in the emergence stage is reorganize the problem descriptions by interface component (Figure 5.8) instead of the original layout by heuristic. The first step in this process is to identify the problematic components of the interface and establish an area to collect all the raw problem descriptions relevant to that component. For each problem element of the interface the users create a text annotation labeling that element and defining a space to hold the related raw problem descriptions. The text annotations are created by simply positioning the mouse cursor over an unoccupied portion of the workspace (R19) and typing in the text used to represent, in this case, the problematic element of the interface. The text annotations can be subsequently edited, freely moved about the workspace, or deleted as the users see fit to best express their emerging understanding.

All the raw problem descriptions around the text annotation are now perceived to be related to the interface component for which it stands (R17). Thus the users have reorganized, and as a consequence reconceptualized, the entire data set. In doing so, they make use of the entire workspace, exploiting the spatial environment (Figures 5.8, 5.9, 5.10). At first (Figure 5.8) the cards are arranged in a haphazard manner around a text annotation. Once all the cards are roughly grouped, the users return to each grouping to impose a more nuance expression of the relationships between the cards grouped in each area. In doing so, they are using spatial proximity (R17) at a more local level to reflect a more sophisticated understanding of what is wrong with that particular element in the interface (Figure 5.9). The users continue to organize at the micro level until they have straightened out the entire workspace. The resulting organization (Figure 5.10) accounts for almost all of the problem descriptions, but is not satisfactory to the users. This is because it does not account for all the cards, and as well “Input” is not a component of the interface. Based on

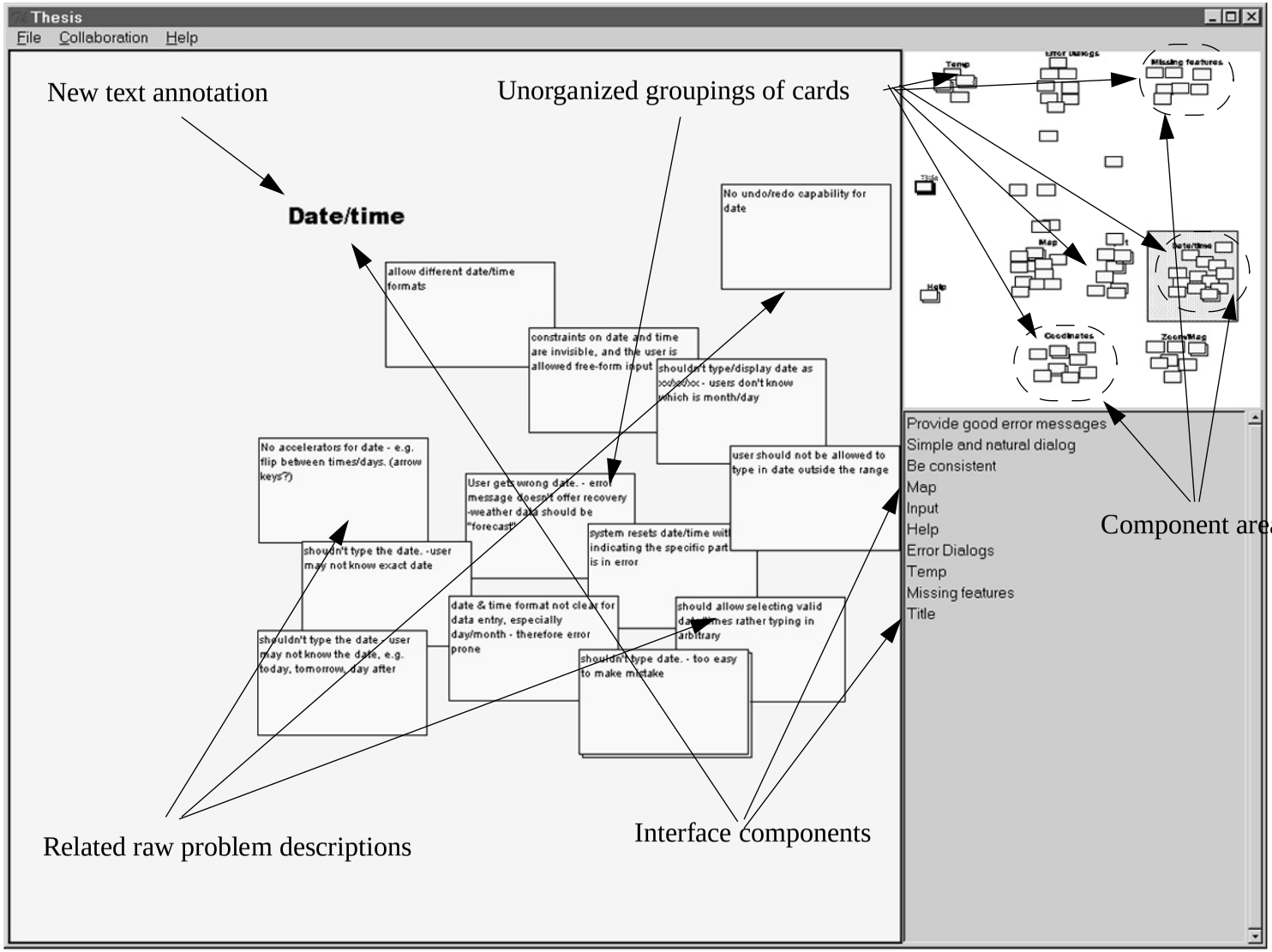


Figure 5.8: Cards reorganized roughly by interface component

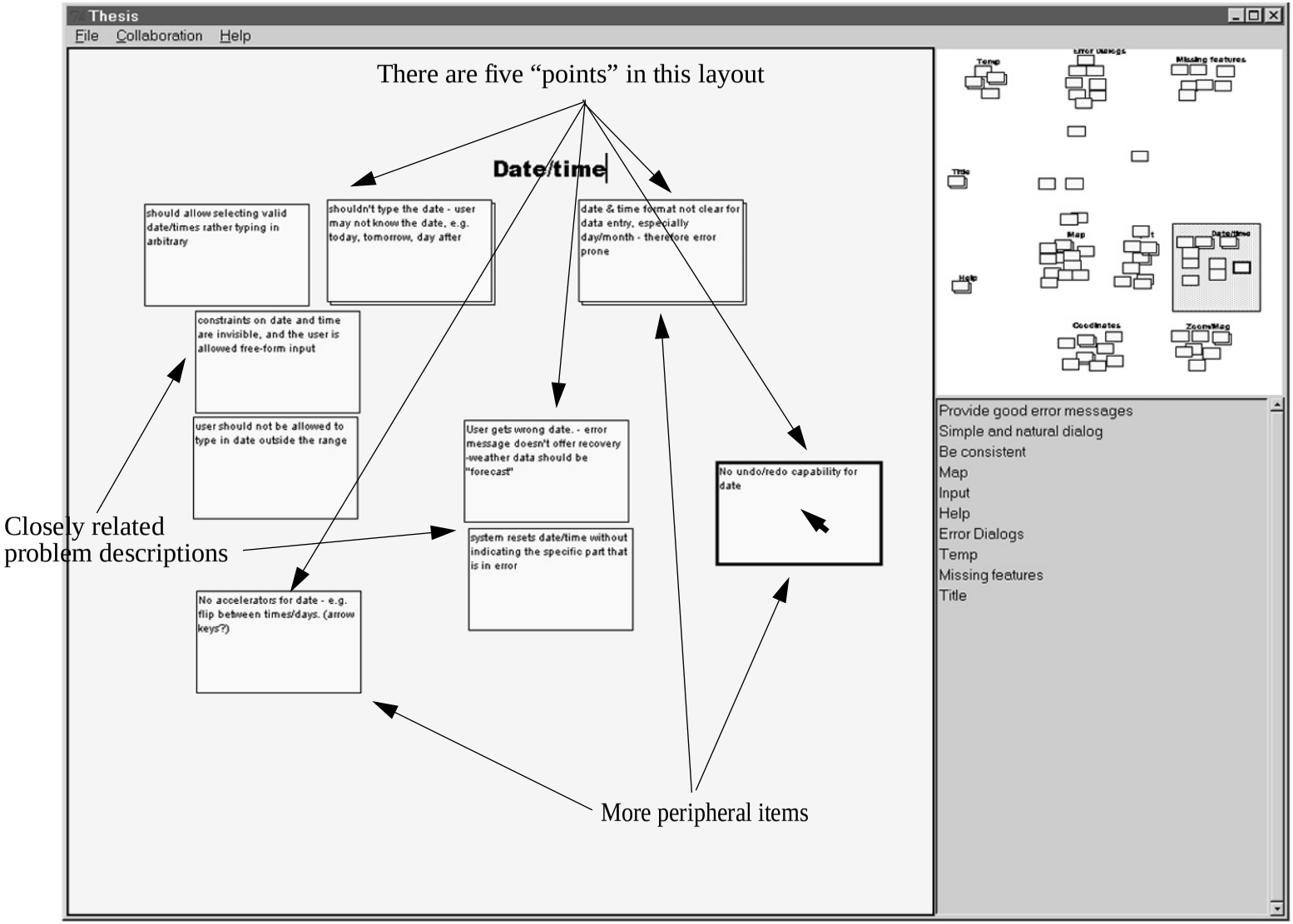


Figure 5.9: Micro-level organizing of a single component related grouping

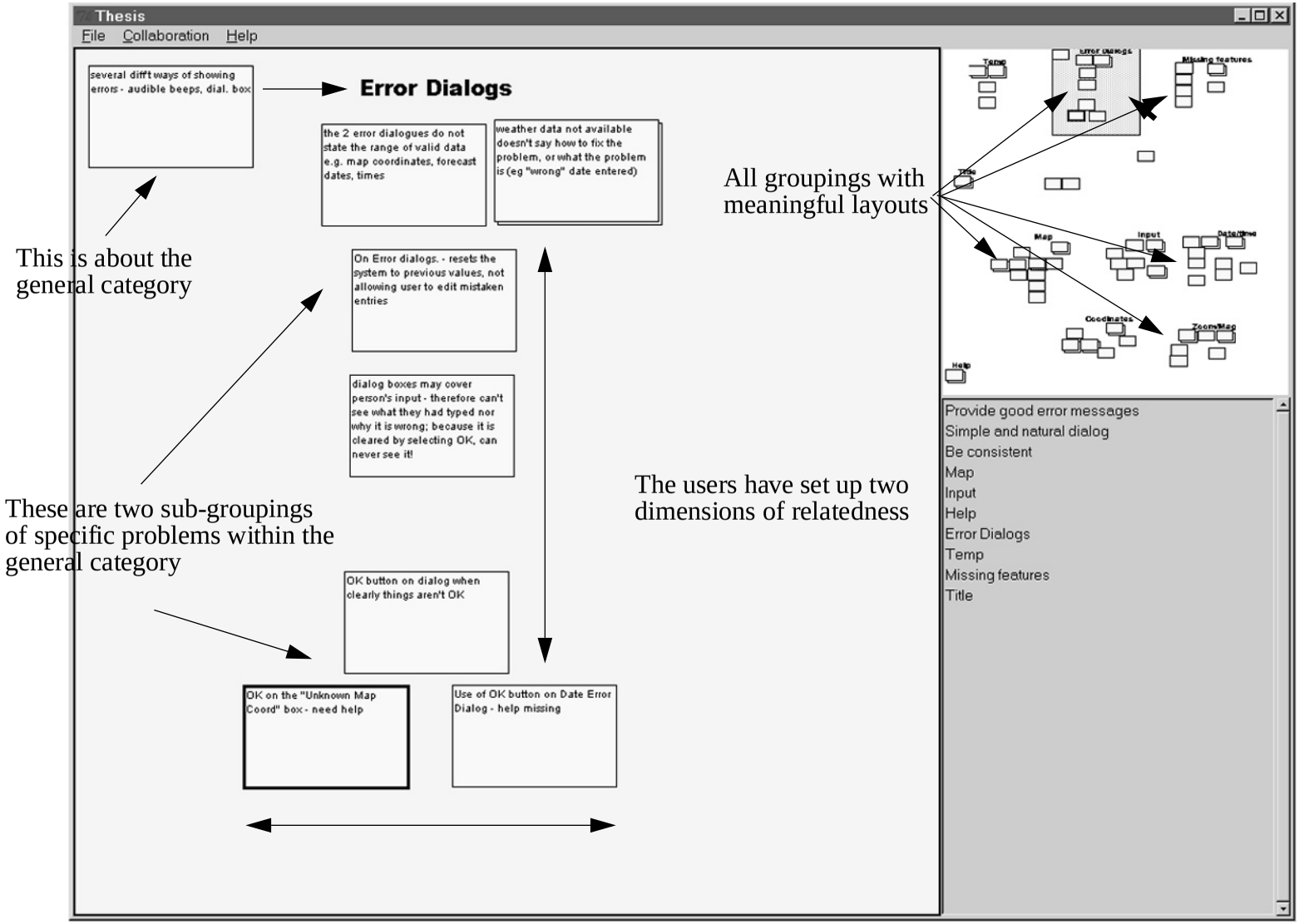


Figure 5.10: All groupings with micro-level organization, first reorganization complete

these observations as well as the further review of the data that occurred in the local organizing of the groupings, the users decide to reorganize yet again, but along different lines.

As in the scenario presented in Chapter 3, the first step taken by the users is to identify the "Help" grouping as one that does not need further work as well as a number of other relatively uncontroversial groupings such as "Title" and "Options." The strategy adopted by the users is now one of "divide and conquer" rather than global reorganization. They are not confident in their agreement on what the complete set of final groupings will be, so they work in closer collaboration, identifying the next grouping and performing the micro level organizing before moving on to identifying the next final grouping. This places an additional demand on the users' use of space within the workspace as they now need some way of keeping track which groupings are those they have decided belong in the final layout and which are still subject to revision. These particular users adopt the strategy of partitioning the workspace into "dealt with" and "not dealt with".

This second round of reorganization represents a more sophisticated understanding of the problems in the interface and a more sophisticated use of the capabilities of PReSS. The new groupings emerge slowly out of recombinations of parts of the previous organization. These new groupings feature a more sophisticated use of spatial proximity, creating multi-level groupings (Figure 5.11). There is a general principle identified ("Make choices visible") that relates to two different parts of the interface ("Date" and "Mag"), with the raw problem descriptions relating to the general concept applied to those particular parts of the interface arranged under the respective annotations. The inspectors have noticed this general concept as well, and the raw problem descriptions arranged alongside the top level annotation address the concept rather than a specific part of the interface. The fact that they are overlapping but not in a pile suggests that they are closely related but substantively different enough to warrant individual consideration in composing the problem reports. These particular users have also wanted to express relationships between this grouping and two others. They have done this with a combination of freehand annotations (R18) and text annotations. Free-hand annotations are created by clicking and dragging

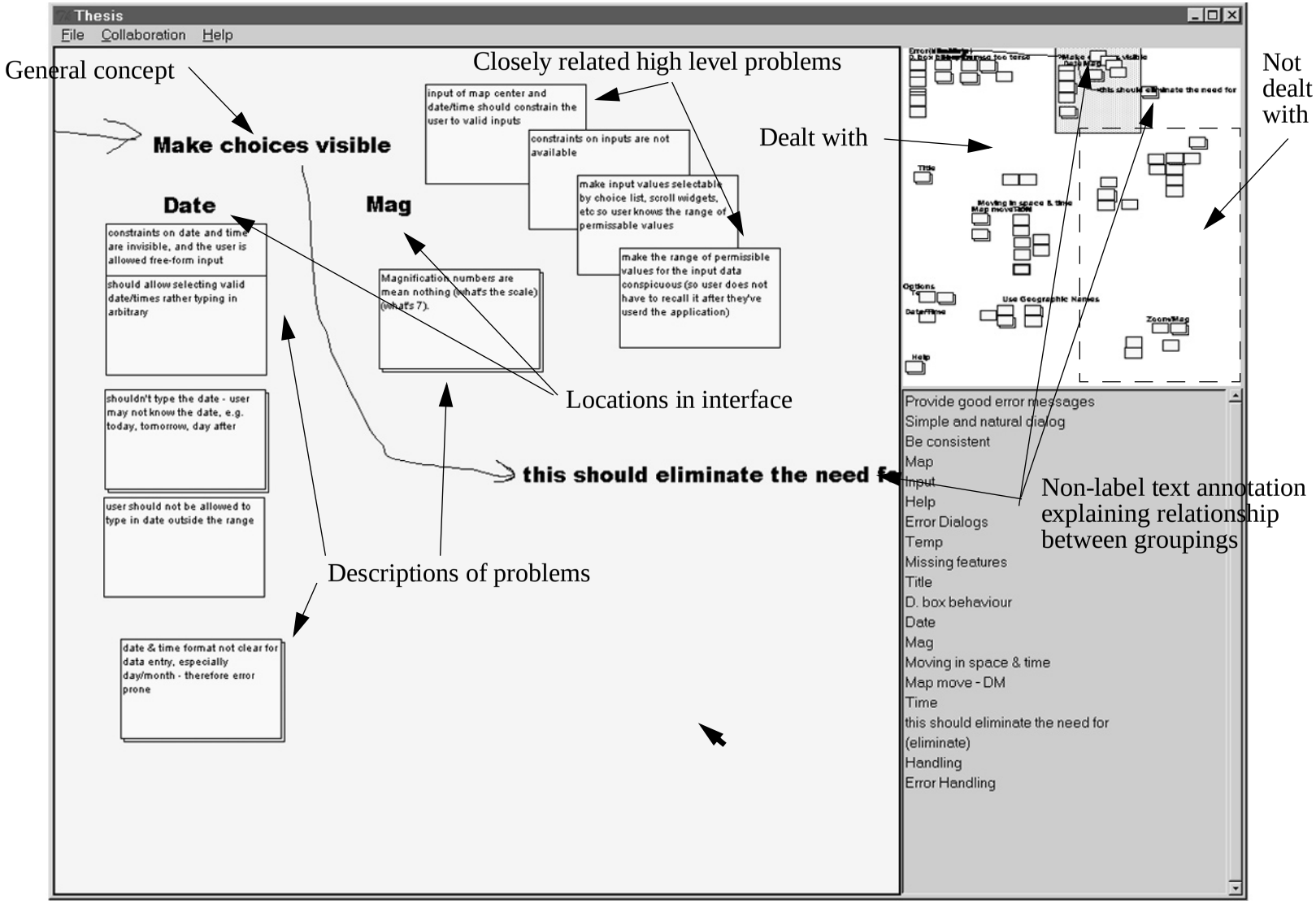


Figure 5.11: Sophisticated use of annotations and spatial proximity

the left mouse button on any unoccupied point in the workspace, much like in paint programs. They can be deleted, but not moved.

Note also that the “dealt with” space has expanded to occupy the left half of the workspace while only a few cards remain in “not dealt with” space. Once the “not dealt with” space has been eliminated (Figure 5.12, compare with Figure 3.11), the emergence stage is complete and the users move on to interpreting their layout. These users have reduced ninety-two raw problem descriptions originally organized in nine heuristic groupings into eight groupings that provide a much better sense of what is wrong with the interface under evaluation and what should be done to fix it.

The astute reader will have noticed that there are a handful of requirements for supporting emergence that have not yet been discussed. These requirements (Table 5.2) have indirect support in PReSS. I believe that the satisfaction of these requirements cannot be ensured solely by system action. Rather it is up to the users to see that these requirements are met, and it is up to the system to do what it can to aid and abet the users in fulfilling the requirements.

Ensuring a Quality Outcome
Completeness
R6: Consider all the operative constraints on the interface.
R8: Consider all the applicable trade-offs in the interface.
Coherence
R11: Create a consistent set of underlying assumptions.
R12: Create consistent language and terminology.
Conciseness
R14: Express problems and their solutions at the right level of abstraction.
R15: Examine ambiguous problems.

Table 5.2: Requirements supported indirectly

The two conciseness requirements (R14, R15) are supported by the affordances of the spatial environment provided by PReSS. Spatial environments are good for expressing

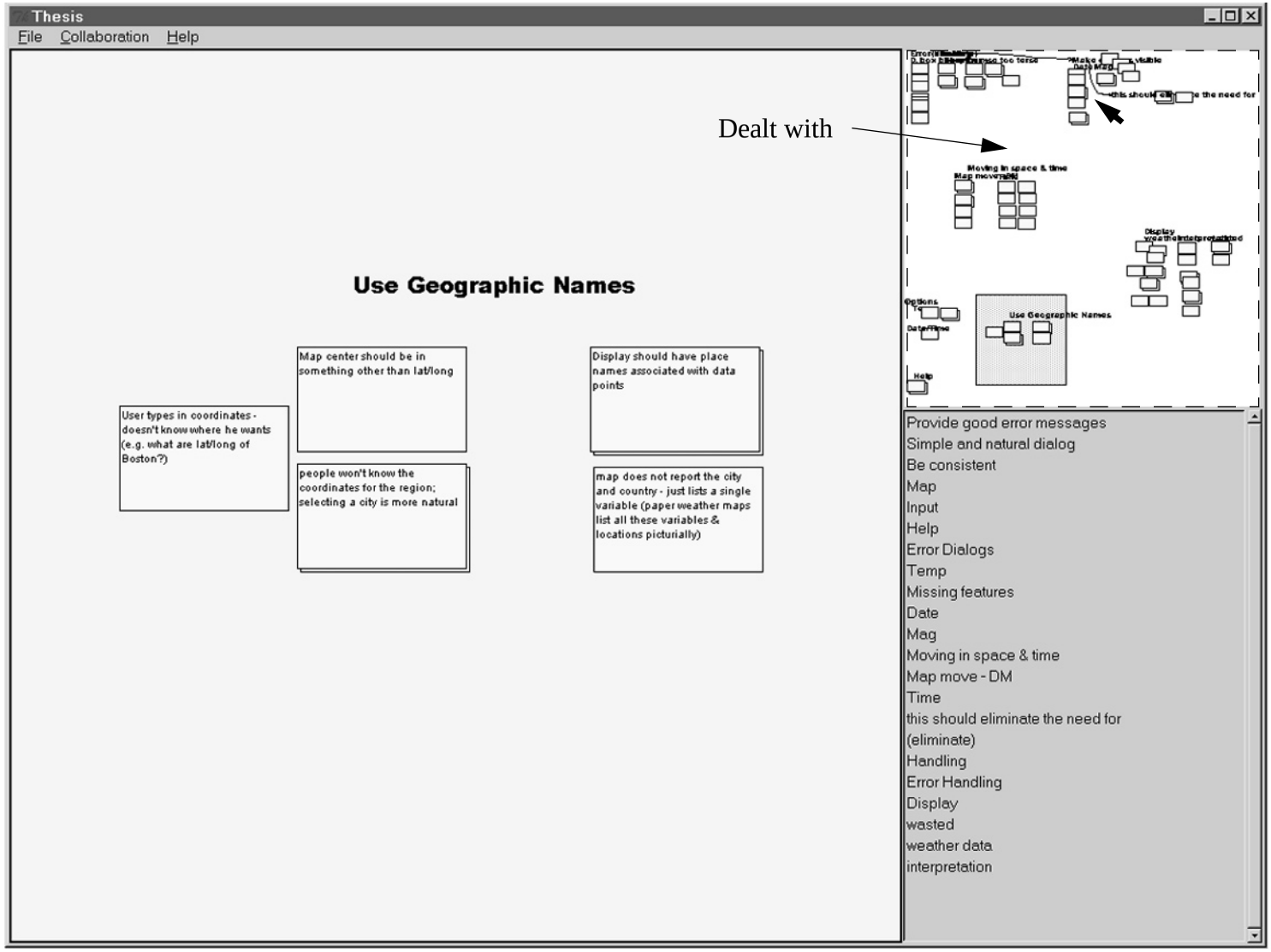


Figure 5.12: Second reorganization complete

ambiguity, uncertainty, and tentative relationships (Marshall, Shipman, and Coombs, 1995). Since it is easy to express ambiguity and since there is no need to explicitly classify problem descriptions until the very end of the process, the users are expected to, in the course of results synthesis, identify ambiguous problems and examine their relationship with other problems and consider what may be done about them. Similarly, spatial layout, in combination with workspace annotations, allow the users to express the different levels of abstraction at which problems may be understood to exist. This can be seen in the compound groupings of cards and annotations shown in the final layout (Figure 5.11). In this layout there are three levels of abstraction apparent:

1. At the most concrete level are the individual cards or piles.
2. A more abstract level is the subgrouping identified by one of the text annotations.
3. The highest level of abstraction is the entire grouping.

What constitutes the right level of abstraction for presenting the problem or problems to developers depends on the context for the evaluation. One factor in the decision is where the project is in its development cycle. If it is early in the development cycle, then a higher level of abstraction is probably best, while if it is late, then a more detailed level of feedback is likely to be more effective in improving the interface. And of course throughout the emergence stage, the users will be arranging the cards into grouping that represent more or less abstract concepts in a fluid fashion according to whatever makes the most sense to them at the time. For example, after reorganizing by interface component (Figure 5.10), most of the groups refer to concrete elements of the interface (See Appendix A). However, the “Input” grouping is more abstract, dealing with behaviour that applies to more than one element of the interface.

Four of the six requirements from Table 5.2 are intellectual activities that must be engaged in by the users collaboratively. PReSS fosters the satisfaction of these requirements by creating a situation where the users have to deal with a common set of artifacts (the raw problem descriptions) that all refer to single entity (the interface under evaluation) and have to be turned into a single, sensible statement (the final problem reports). In order to compare the problem descriptions and discover the relationships amongst the con-

tributions from different inspectors, the users will have to develop a common vocabulary (R12) for talking about the interface, its problems, and their causes and solutions. Further, differing assumptions will lead to different problems, different descriptions of the same problem, or different solutions to the same problem. Results synthesis, performed correctly, will lead to the identification of these differences, and to their resolution because remaining differences will prevent the participants from agreeing on how to best express the problems in the interface nor on how to best address the problem. In the same way, the discussions that arise amongst the participants in the course of dealing with the problem descriptions and creating the final problem reports will lead to identifying the constraints (R6) on and the trade-offs (R8) in the design space.

5.6 Finalizing

When the participants have come to agreement about the problems in the interface and all the raw problem descriptions have been accounted for, the emergence stage has been completed and it is time for the last stage in results synthesis. This stage involves translating the knowledge encoded in spatial layouts and annotations into a form that is more easily communicated to others in the development process. In PReSS, this is supported by providing a problem report element that is created in the workspace itself. I expect each grouping in the workspace will have at least one associated problem report. However, it is possible that the participants will decide to create more than one problem report for larger and more complex groupings.

Problem reports are created by command from the workspace pop-up menu. The problem report (Figure 5.13) has three sections, presented with a tabbed window control, with one section for a description of the problem (R21), one for a description of the solution (R20), and one for an indication of the severity of the problem (R22). The problem reports are created as elements in the workspace to help the users keep track what groupings they have dealt with and what remain to be considered. This also enables the other elements in the workspace, the cards and annotations, to serve as context and detailed reminders of what belongs in the problem report.

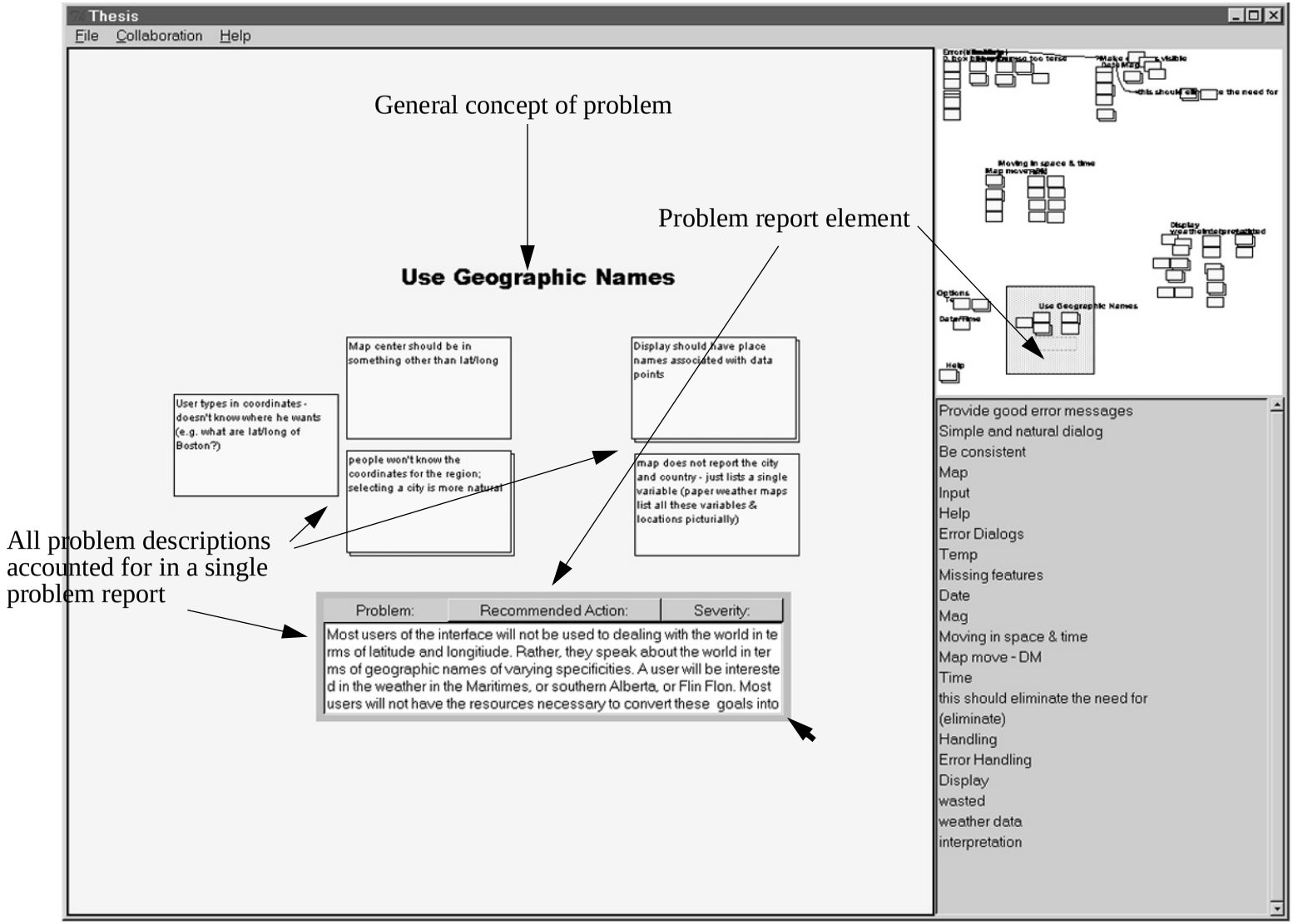


Figure 5.13: Problem report created for "Use Geographic Names" grouping

The problem report can be freely moved about or deleted. The different sections of the problem report are accessed through the buttons on the top of the report. When the *Publish* command from the **File** menu is invoked, an HTML page is created for each problem report in the workspace. Figure 5.14 shows the HTML page created for the problem report shown in Figure 5.13.

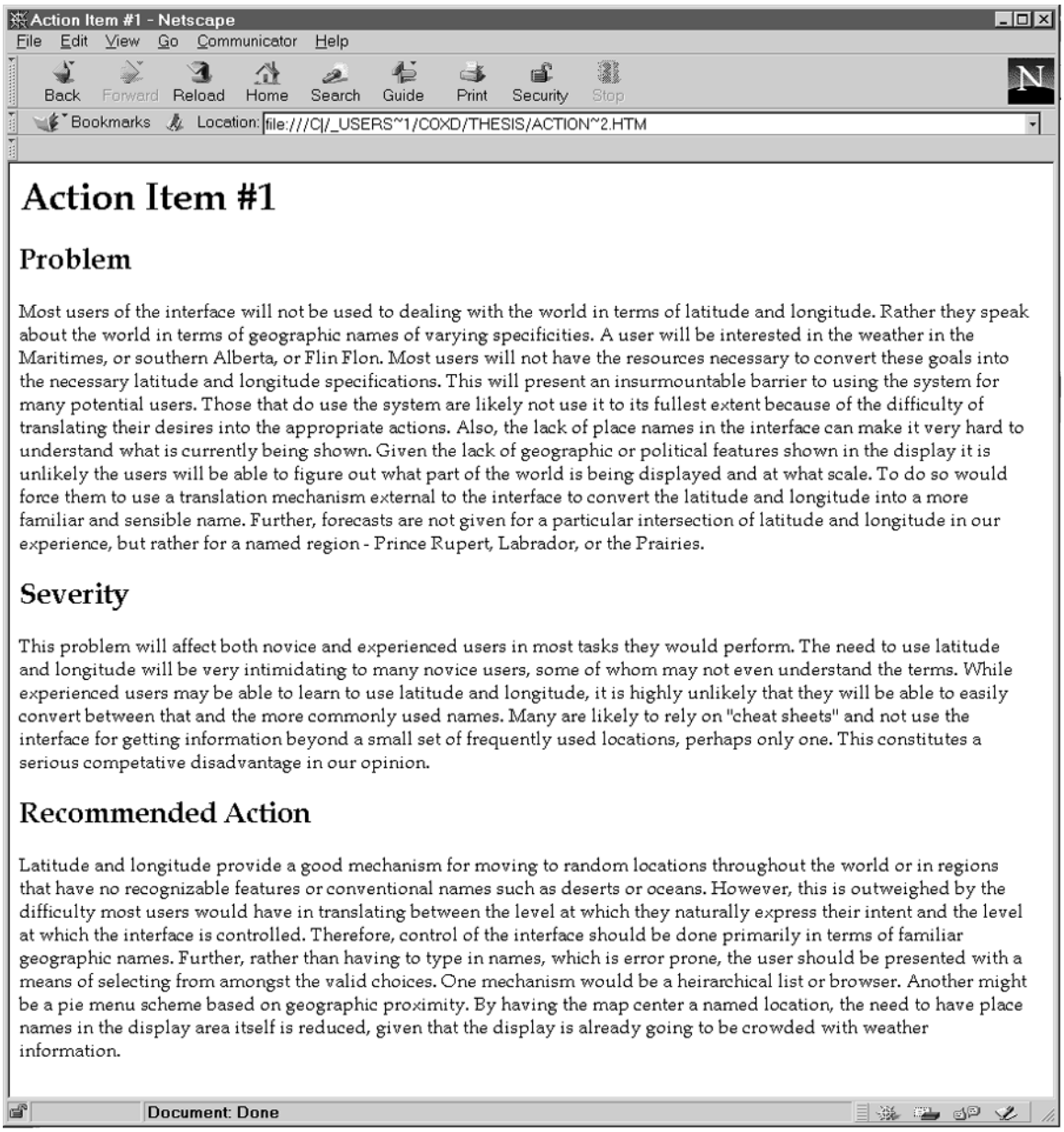


Figure 5.14: Web page generated from final problem report for “Use Geographic Names”

5.7 Conclusion

This chapter has discussed the key design points for a system to support results synthesis. Table 5.3 summarizes the design response to the requirements put forth in Chapter 4. The system is based on a two dimensional workspace that users interact with through a main view and an overview. The design of PReSS is aimed at mimicking the strengths of paper based environments with further enhancement through the power available in computer representations. The users' actions consist of rearranging cards representing problem descriptions, and adding and rearranging annotations in the workspace. The system fits efficiently into the development process by enabling the easy procurement of input and the capturing of results in a form that can be used directly in the next step in the development process. There are some requirements that the system supports indirectly by providing an environment conducive to the participants achieving these requirements.

Requirements	Design Response
Elements of Raw Problem Descriptions	
R1: Include a description of the problem.	Take input from a system designed to solicit a problem description.
R2: Include the associated heuristic.	Take input from a system designed to record an associated heuristic.
R3: Provide the inspectors with a space to record possible solutions.	Take input from a system designed to allow for the recording of solutions.
R4: Include an identification of the author.	Not currently supported.
Ensuring a Quality Outcome	
Completeness	
R5: Consider the entire collection of raw problem descriptions.	Make the entire collection of raw problem descriptions visible at once in the overview. Define the process to include a review of the entire collection.

Table 5.3: Summary of design from requirements

R6: Consider all the operative constraints on the interface.	Provide an environment that makes it easy for the participants to discuss constraints and facilities for recording identified constraints.
R7: Provide adequate justification for every problem.	Provide space for recording problems.
R8: Consider all the applicable trade-offs in the interface.	Provide an environment that makes it easy for the participants to discuss trade-offs and facilities for recording identified trade-offs.
R9: Generate workable solutions to the problems raised.	Provide space for recording solutions in the final problem reports.
R10: Generate alternate solutions for problems where possible.	Provide space for recording alternate solutions in final problem reports.
Coherence	
R11: Create a consistent set of underlying assumptions.	Provide an environment that makes it easy for the participants to discuss assumptions and a facility for recording agreed upon assumptions.
R12: Create consistent language and terminology.	Provide an environment that makes it easy for the participants to identify varying uses of language and terminology and a facility for recording agreed upon usage and meaning.
Conciseness	
R13: Eliminate duplicate problem reports.	Pile mechanism to collapse duplicate problem reports into a single element.
R14: Express problems and their solutions at the right level of abstraction.	Provide an environment that enables the expression of the relationships amongst different levels.
R15: Examine ambiguous problems.	Provide an environment that allows the identification of ambiguous problems.

Table 5.3: Summary of design from requirements

Enabling Emergence in Results Synthesis	
R16: Provide a spatial environment.	The system is based on a spatial metaphor, and provides a large space in which all activity takes place. An overview is also provided to mitigate the constraint of using a physically small display to view a conceptually large space.
R17: Use spatial proximity and visual cues to express relationships amongst the data.	Spatial proximity and visual cues are the default mechanisms provided for expressing relationships amongst the data. The system displays these two relationships on two levels: fine detail in the main view and global context in the overview.
R18: Allow free form annotation of the underlying space.	Freehand drawing is provided in the workspace, and text annotations may be created anywhere.
R19: Allow the free creation and movement of data in the space.	Annotations may be created at any time without entering a mode. Most elements in the workspace may be moved to any location by dragging.
Recording the Outcome	
R20: Document the outcome of results synthesis.	Problem reports are created in the same space as the content on which they are based. Final problem reports are “published” as a set of HTML pages.
R21: Provide separate accounts of problems and their solutions.	Final problem reports have separate sections for describing the problem and the solution(s).
R22: Include a description of the severity of each problem.	Problem reports have a separate section for severity that allows for more verbose descriptions of the problem severity than a single numeral.

Table 5.3: Summary of design from requirements

Who Performs Results Synthesis	
R23: Results synthesis is participatory.	Keep the system simple and make it groupware enabled so that diverse audiences in different locations may participate in results synthesis, eliminating the barriers of setup and travel costs otherwise required.
R24: Include the inspectors in results synthesis.	Import raw problem descriptions without requiring further processing.
R25: Include developers in results synthesis.	No explicit support provided.
R26: Include end users in results synthesis.	Keep the system simple.
R27: Systems supporting results synthesis are groupware enabled.	Implement system using GroupKit groupware toolkit.

Table 5.3: Summary of design from requirements

Chapter 6: Evaluation

6.1 Introduction

In this chapter I present my evaluation of PReSS as presented in the previous chapter.

System evaluations, particularly those focusing on the system interface, can be roughly divided into two types: *formative* and *summative* evaluations. Summative evaluations attempt to evaluate a system against some criterion to judge whether or not it has achieved its developers' goals. Formative evaluation aims to guide the evolution of the design of the system by providing feedback to the designers on their efforts up to that point. The system I designed is exploratory in nature, so I believe that a summative evaluation would be premature and cannot be justified given our current level of understanding of the underlying process and how to design systems that support this process. Consequently, the rest of this chapter will describe a formative evaluation of my system for supporting results synthesis (PReSS). The end goal of this process is not to explore many different alternatives nor to develop the "best" interface, but to see if the proposed design is reasonable and if so to create and refine it as a baseline usable system amenable to further research and development.

In Section 6.2 I describe in detail the method I used in performing the formative evaluations. Subsequent sections describe the actual studies in chronological order. The first series of studies I ran involved only a single user operating the system. These studies are described in Section 6.3. The goal of these studies was to drive out software defects and assess a basic level of usability in PReSS without the complexity of group operation. This was achieved, with significant changes made to the interface presented in Chapter 5 as a

result of the evaluations. Having reached a point of diminishing returns in the single user trials, I proceeded to a series of multi-user trials which are described in Section 6.4. I conducted five multi-user studies, four using two users and one with three users. These trials demonstrated that the system was usable in a groupware situation. While some shortcomings in the system were noted, they did not result in substantive change to the interface. In Section 6.5 I summarize the results of this study and present a brief critique of the evaluation.

6.2 Method

The method chosen for this evaluation is observational usability studies for the purpose of iterative, formative evaluation. Unlike traditional controlled experiments, this evaluation method does not identify independent or dependent variables. Nor is there any notion of controlling the variables, treatment levels, or concern about confounding. Rather, after each iteration the observations of the experimenter are used to refine the system under evaluation as well as the study itself and the selection of future participants. This refinement reflects the experimenters' current understanding of the problems to be solved and ability to make changes that the experimenters think will have a positive impact. Thus, it is a highly qualitative and subjective method. As such, it is more likely to detect large problems – places where there is substantive dissonance between the intention of the design and the reality of its use. More nuanced and ambiguous deficiencies in the design are less likely to be detected.

I chose this method because the goal of my research with respect to the system is to show that it is usable and useful for results synthesis. The interface is relatively simple and

designed from strong theoretical motivation. The focus of the evaluation is confirming that the system can be used to successfully perform results synthesis and further refining the system to better support the process. I felt that because the process and design were still largely speculative, a more summative or comparative approach to evaluation was not appropriate. I expect that as more experience is gained with results synthesis, more rigorous evaluations will be undertaken.

The evaluation is divided into two stages: single user studies and multi-user studies. Studies with a single user were done first to drive out software defects and establish a base level of usability. The single user scenario is much easier to observe and analyze than multi-user situations. Groupware usability is notoriously hard to evaluate (Grudin, 1988; Gutwin, 1997). However, what makes a good interface for an individual user may not make good groupware (Gutwin and Greenberg, 1998). Consequently I embarked on the single user evaluations with a certain amount of caution. My goal was to stop the single user trials and move on to multi-user trials as soon as system operation was mostly defect free. In addition, I expected that all identified usability problems would be repaired where such repairs were not expected to violate groupware usability. The single user trials are reported in more detail in Section 6.3.

Following the single user trials, I performed a number of multi-user studies aimed at discovering additional usability problems with the system operating as real-time groupware. These studies also sought to determine whether in fact the system could successfully and reasonably be used in the multi-user situations. The issue of how well the system

would scale as more participants were active on the system was also looked at. These studies are reported on in more detail in Section 6.4.

The structure of each study was the same, regardless of whether it was multi-user or single user. The participants were given a brief demonstration of the system's capabilities, after which they used the system to perform results synthesis. They were limited to one hour on task. This was followed by a retrospective interview. I would then create a list of what changes I thought should be made to the interface based on my observations. Appendix E contains the summary reports for each study I conducted. I would implement those changes I identified before the next study and use the updated interface in subsequent studies.

In all studies, the participants performed results synthesis on a set of problem descriptions that were generated by inspecting the interface described in Exercise 8 in Nielsen's *Usability Engineering* (1993, p. 273-4) (See Appendix A). Two different data sets were used. Each was generated by a different group of five inspectors. One group generated ninety-two problem descriptions, the other eighty-four. One reason for the two data sets is that I tried to have the participants perform results synthesis on a data set to which they had contributed. This is because my conception of results synthesis is that the participants include the inspectors. This approach was borne out in the studies because there was a noticeable difference in behaviour between those who had contributed to the data set and those who had not.

Study – User	Data Set	Contributed to data set	Results Synthesis Experience
1 – 1	DS1	Y	Y[a]
2 – 1	DS1	N	Y
3 – 1	DS1	Y	Y[a]
4 – 1	DS1	N	N
4 – 2[b]	DS1	Y	Y[a]
5 – 1	DS2	Y	N
5 – 2[b]	DS2	N	Y[a]
6 – 1	DS2	Y	N
6 – 2	DS2	Y	N
7 – 1	DS2	Y	N
7 – 2	DS2	Y	N
8 – 1	DS2	N	N
8 – 2	DS2	N	Y[a]
8 – 3[b]	DS2	N	Y[a]

Table 6.1: Study participants experience and relation to data set used.

a. Performed paper-based results synthesis with data set DS1.
b. The experimenter

The first data set was taken from my first study of results synthesis in a paper-based environment. Participants from that study also took part in a number of the system evaluation studies, including the experimenter. The primary reasoning for this reuse of participants is a principled devolution of the ability of users of PReSS to cope with deficiencies in the system. Within each type of study, I sought to start with participants who were familiar with the process so that any problems that arose could be attributed to either soft-

ware defects or deficiencies in the design of PReSS. As I became more confident in the robustness of the system, I would move to using participants that were not only unfamiliar with the interface, but who also had no previous experience with the results synthesis process. Table 6.1 summarizes the previous experience of the participants in each study and their relationship to the data being used in the study. Where appropriate, informed consent was requested of and provided by subjects.

6.3 Single user studies

I performed three single user studies. The focus of these studies was to identify software defects and establish a base level of usability. The single user studies were carried out on PCs running Windows NT™. The rest of this section will report in detail on each study.

6.3.1 Study #1 – Single user

The participant was a male Computer Science professor specializing in groupware and HCI. The participant was a contributor to the data set, and had previously performed results synthesis in a paper-based environment on this data set (Table 6.1). However, it was over a year since the participant had taken part in that study. The participant was intimate with my research on results synthesis as well, and had contributed to the discussion of it. The participant used the first version of the interface as described in Chapter 5. In the allotted time, the participant did not complete the final stage of results synthesis, but made significant progress and said that results synthesis would have been completed given more time.

The participant was able to use the system in the manner envisioned in its design for the most part. The participant used all of the views in the interface. However, a number of

differences occurred between results synthesis carried out in a paper-based environment versus using PReSS. One such difference observed in this study was that while spatial layout was used, it was not exploited to the degree or with the same subtlety seen in the studies of paper-based results synthesis. The system is functionally capable of accommodating the extensive and nuanced use of spatial layout as was seen in the paper-based environment. However, the participant chose, consciously or not, to use the system in a different way. The user used piles in a much broader manner than I anticipated based on what I observed in the paper-based environment. The user used piles to collapse many cards together, rather than spreading them out as was typical in a paper-based environment. I had envisioned the piles only being used for exact duplicates. I believe the reason for this that PReSS users placed a higher priority on reducing the perceived complexity of the data set.

Indeed, the desire to reduce apparent visual complexity by moving quickly to a more compact representation was a theme that emerged in this and later studies. There was a general desire to reduce "clutter" – to simplify the data representation – with all due speed. Participants would often express concern when first confronted with the large study data set. They viewed it as very complex, and in some sense intimidating. Their first instinct was to reduce this complexity so that they were more comfortable operating in the environment. It was a strategy to reduce cognitive load.

At a more mundane level, a number of software defects were discovered and corrected before the next trial. Three changes were made to the interface as a result of this study. Participants often wanted to search the workspace for a problem description they remem-

ber seeing before that was related to one currently in their main view. In the original design, they would have to move their main view in order to look for the remembered card, which might lead to them losing track of where they were, leading to disorientation or frustration. This led to the addition of an *info area* (Figure 6.1). The info area displays the content of certain items when the mouse cursor was positioned over them in the overview. The info area shows the content on cards or the top card in a pile. The info area allows a user to leave their main view at the position of interest while they search for the problem description that is no longer in the main view using the combination of the info area and the overview. This facility is especially useful when the user remembers that the card they are seeking was a member of a particular group, but not its exact location within the group.

Next, the participant often created freehand annotations accidently when attempting to select elements in the workspace. This problem could have been predicted since selection using the left mouse button is a common binding in most graphical user interface applications. Thus the second change was to make group selection the default action for mouse button 1 and require a modifier key to create freehand annotations.

Finally, it became apparent in this study that piles were being used in a way that was not envisioned in the original design. In the original design, they were to be used only for collapsing duplicates. However, they came to be used as a more general grouping mechanism, bringing together problem descriptions that all related to “the same problem.” This notion of what constituted the same problem was something that emerged as part of the process. Consequently, I felt it was useful to support the merging of piles, as the user may

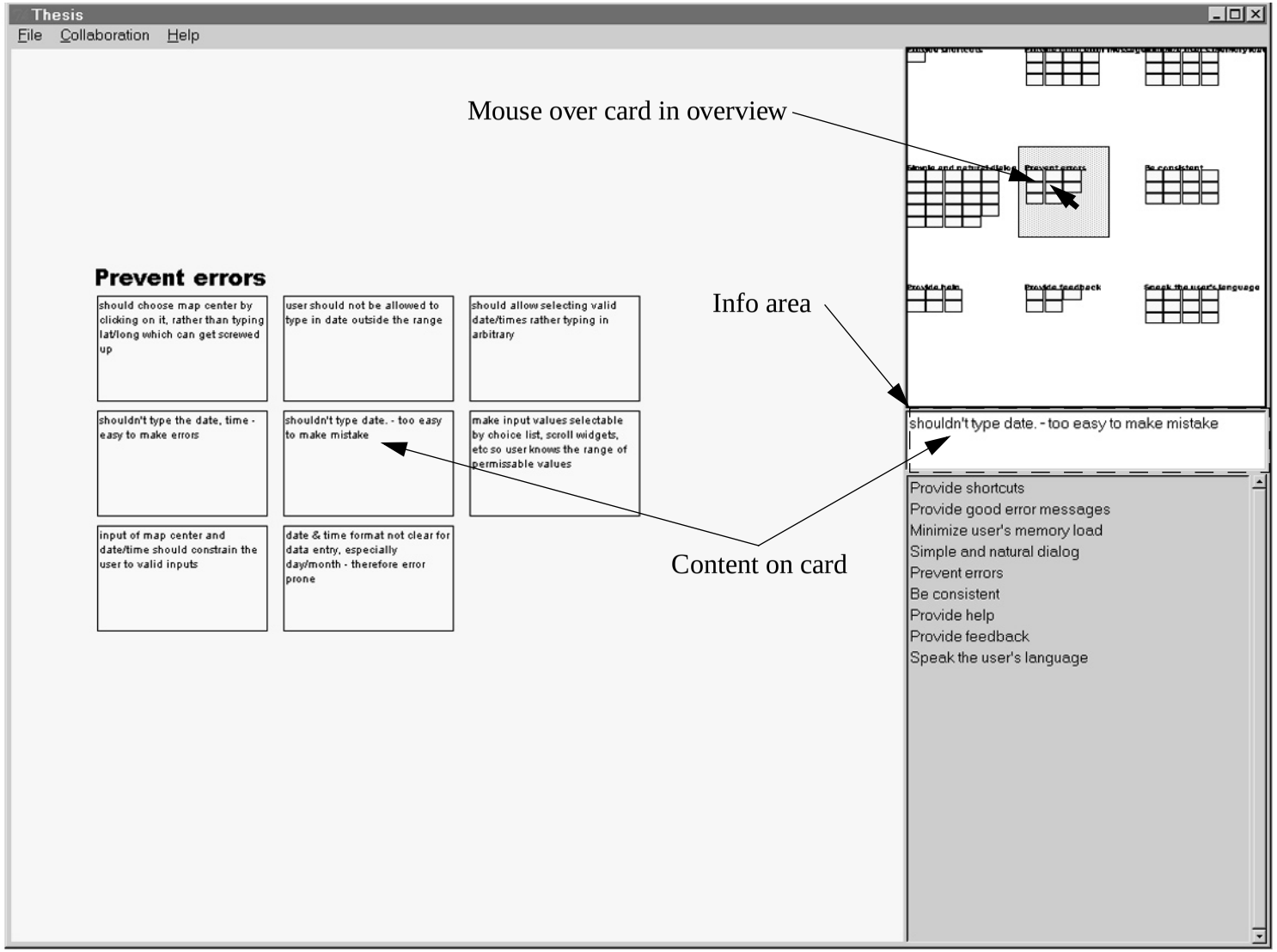


Figure 6.1: Info area added

in the course of performing results synthesis come to the conclusion that what he or she thought were separate problems, represented by separate piles, have come to be seen as the same problem, and hence should be a single pile. The third change to the system was to add the ability to add a pile to a preexisting pile.

6.3.2 Study #2 – Single user

The participant was a male graduate student in Computer Science with HCI training and industry experience as a member of a Human Factors group in a local company. The participant was not a contributor to the data set. However, the participant was familiar with results synthesis, having performed it in a paper-based environment (Table 6.1). In that instance, the data set being used was different from the one used in this study as a different interface was being evaluated in that instance. The participant did not complete results synthesis within the allotted time, but indicated that given more time he believed that the task could be completed.

As with the participant in the first study, this participant's early activity in the workspace was aimed at reducing complexity. The participant created piles of cards that were not necessarily duplicates but were seen as "the same problem," similar to what happened in the first study. Nonetheless, the participant exploited the interface in many expected ways, employing spatial layouts, moving the cards about the workspace and creating new groupings. The participant also made extensive use of the info window implemented as a result of the first study. I also observed that the participant spent a noticeable amount of time searching in the workspace, moving the main view sometimes but mostly using the

info area, over the course of the study. I believe that this is due at least in part to the fact that the participant was not a contributor to the data set.

A number of software defects were identified and repaired before the next trial. Two changes were made to the system as a result of this study. The first change was to allow users of the system to add content to the workspace that was at the same level as the problem descriptions, yet distinct from them. In this study and the previous one the users asked about ways to add new content separate from text annotations. While users could previously add content through text annotations, this was not exploited in the same way as seen in the studies of paper-based environments presented in Chapter 3. In those studies, I observed participants annotating the work surface to record additional notes about the problem descriptions – text that was not a label for a problem description grouping (See Figures 3.10, 5.12). In PReSS, the way the text annotations were displayed and the fact that their appearance could not be modified led users to avoid using them for adding non-label text annotations. This change was effected by the addition of a *note* element to the workspace that allows users to add content to the workspace. Figure 6.2 shows how notes appear in the main view and the overview. The info area will show the note content when the mouse pointer is over a note in the overview.

Next, I noticed that this and the previous users were having trouble remembering all the mouse and key bindings present in the system. I thought that by making the bindings more consistent, it would make them more memorable as well as mutually reinforcing. The second change was to change the mouse button used for panning in the overview (by

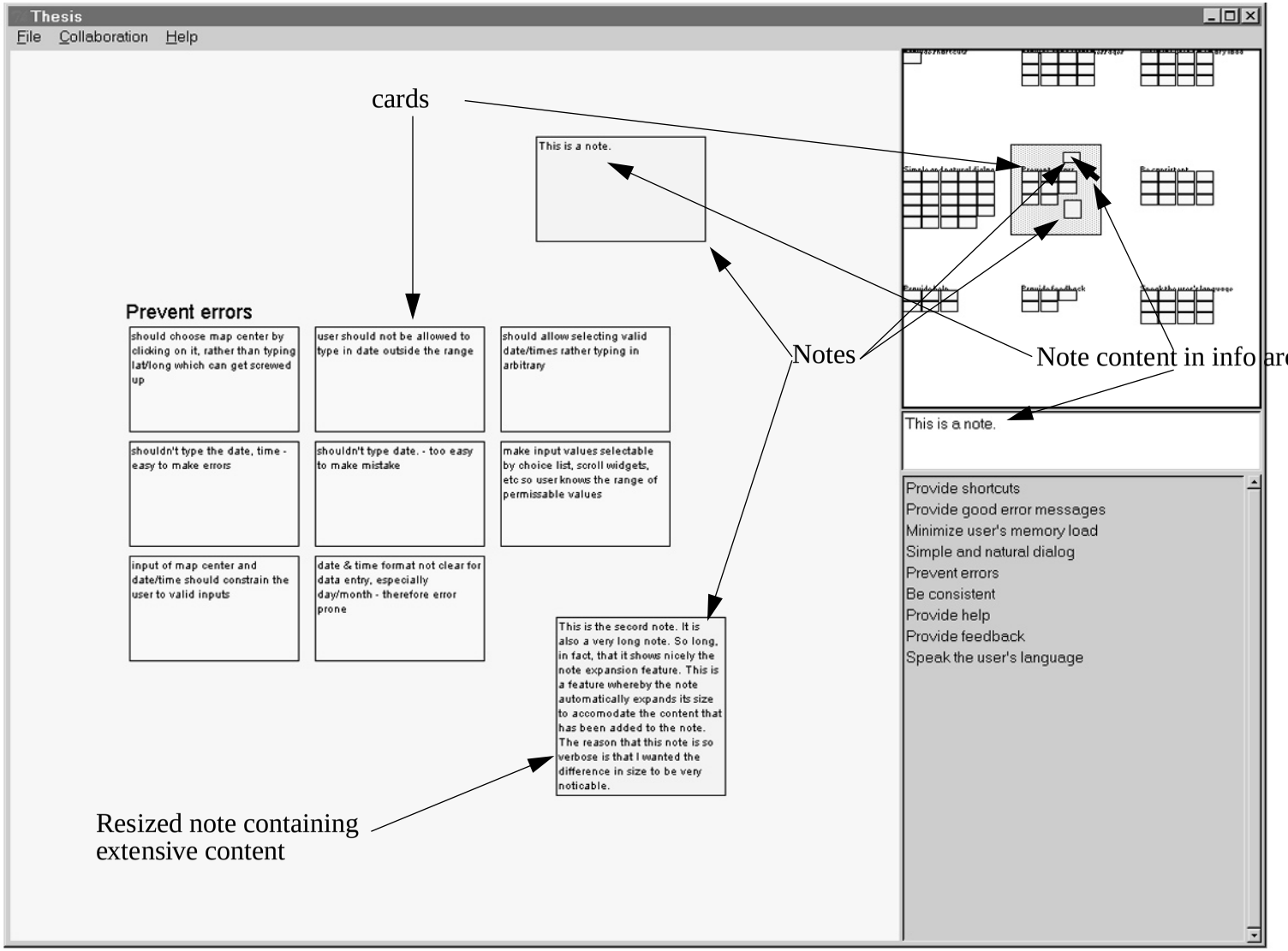


Figure 6.2: Note elements added to the workspace

dragging the view rectangle) to be the middle mouse button in order to be consistent with the bindings in the main view.

6.3.3 Study #3 - Single user

The participant was a male software developer who had a M.Sc. in Computer Science and HCI training. The participant was a contributor to the data set. The participant had previously done results synthesis on the same data set in a paper-based environment in one of the studies reported in Chapter 3 (Table 6.1). As in the previous studies, the participant did not complete the results synthesis in the allotted time, but indicated that the task was completable given more time. The participant had a background in software development and had very particular ideas about the best way to report problems. This led to the participant getting the furthest along in the results synthesis process. However, this speed was at the expense of deeper consideration and openness to alternative organizations. The participant did not exercise as much of the functionality of the system as other approaches might have.

In spite of the participant taking what I would consider an unorthodox approach, the participant found the system to be usable for carrying out his streamlined version of the process. The participant created problem reports very early in the process, almost from the beginning. The participant had very definite ideas about what the right level of abstraction was for expressing problems and exhibited little uncertainty in organizing the data. The participant still used many of the features of the system. Spatial layout was used; the participant discovered new groupings that crossed heuristic boundaries; and he was able to

execute the reorganization. The participant also used piles to group cards by the “same problem” metric as was seen in the preceding two studies.

A few more software defects were identified in the study and fixed before the next trial. One change to the interface was made. When trying to create a text annotation, the participant often looked for feedback indicating that he could begin entering the content for the annotation. Previous experience with other systems had lead him to look for an insertion cursor as the sign that he could successfully enter text, but the cursor did not appear as PReSS used a point-to-type model: the user had only to type to create a text annotation at the position of the mouse cursor whenever the cursor was over an unoccupied location in the main view of the workspace. The participant would click repeatedly in the workspace in an attempt to get this feedback. By moving to a click and type interaction, the user is provided with positive feedback (the insertion cursor) that the system is ready and able to accept the content for the annotation.

6.4 Multi-user studies

At this point I decided to halt single user studies and move to performing multi-user studies. A number of software and design defects had been identified and resolved. The system was reasonably robust and users ran into error messages very infrequently. The interface proved to be usable in the single user scenario, with the participants able to perform results synthesis in a satisfactory fashion. A number of improvements to the interface arose as a result of the single users studies, most notably the info area.

The majority of the multi-user studies involve only two participants. I made the trade-off in favour of doing more iterations instead of having more “realistic” situations with

more participants. The two-user case is chosen as a reasonable starting place for multi-user trials as it is the minimum level of additional complexity that will also stress the groupware features of the system. More users can be added to determine if the systems scales well to increased demand once it has been proven in the minimal case.

In studies 4, 5, and 8 I was one of the participants. These were “pilot” studies, aimed more at finding defects in the software or the study itself. While I tried to minimize my influence on what happened, it is inevitable and undeniable that I did exert influence on how the system was used and how results synthesis was carried out. I also found it hard to be both observer and participant and I believe that my observations suffered in both quality and quantity as a result. Studies 6 and 7 used pairs of students from a graduate course in HCI research methods.

The study environment was a simulation of remote interaction. It was set up so that the participants, who were co-located in the same room were facing away from each other and could not see each other’s screen. Participants were encouraged to talk with each other during the task. The aim was to simulate using the system in a remote collaborative setting where the participants had high quality audio conference capabilities but no video connection. The two user trials were carried out on PCs running Linux, while the three user study used a Sun workstation and two PCs running Linux.

6.4.1 Study #4 - Two user

There were two participants in this study. One was the experimenter (P1), who had contributed to the data set. The other participant (P2) was a female Ph.D. student in Computer Science who had HCI training and experience in using real-time groupware. P2 had not

contributed to the data set. P2 was not familiar with results synthesis or the interface on which the data set was based. Initially, P2 did not understand the exact nature of the task. After a bit of explanation from the experimenter, P2 demonstrated a grasp of the goal of results synthesis by participating appropriately and displaying initiative in identifying new groupings and contributing to those initiated by the other participant.

As with the previous studies, there was not enough time allotted in the study to allow the participants to complete the task. The system functioned according to expectation. *Cloning*, the ability to make copies of cards, was used a fair bit, though interestingly the clones tended to end up in the same final group, leading to "extra" instances of the card that had to be put into piles. The participants were able to work apart and together and move between the two without much overhead. The participants made and were able to interpret diectic references to elements in the workspace such as "What do you think of this one <indicating a particular card>". Spatial layout was used.

One change was made as a result of this study. With the click to type model of interacting with text annotations, users were able to directly enter the edit mode for text annotations. Hence I removed the Edit entry from the text annotation pop-up menu as it was identified as superfluous by the non-experimenter participant. A number of software defects were identified and fixed before the next study.

6.4.2 Study #5 - Two user

One of the participants in this study was the experimenter (P1) who did not contribute to the data set, but had reviewed it. The other participant (P2) was a female graduate student in Computer Science enrolled in her first graduate class in HCI. P2 had contributed to the

data set. P2 had never performed results synthesis before. The participants did not complete results synthesis in the allotted time. An initial reorganization had been almost completed when time expired.

The participants used spatial layout, and many of the features of the system were used as with the previous studies. The participants were able to work apart and together which is a property of good groupware design suggested by Gutwin (1997). Diectic references were made and understood by both participants, another property of successful groupware design (Gutwin, 1997). There was a definitely observable learning curve in P2's interaction with the system, both in terms of understanding the results synthesis process and figuring out how to operate the system. Nonetheless, significant progress was made and in a relatively short period of time P2 had developed enough proficiency to contribute substantively to the process.

An interesting note is that a compensatory behaviour developed to deal with P2's lack of skill in moving her main view early in the task. When the participants started to talk about an card that was not in the other participant's view, the initiator would place the element in the other participant's main view so they wouldn't have to navigate to the "original" location. This adaptive behaviour became non-adaptive as P2 became more comfortable with the interface and skillful in moving her view. The earlier behaviour of placing problem descriptions in the other participant's view for consideration remained, occasionally causing some consternation when the other participant was about to move their view and then the object being discussed showed up in their view.

One software defect was observed, but it could not be reproduced in subsequent testing. No changes to the interface were made as a result of this study.

6.4.3 Study #6 - Two user

One participant was a female then employed in the Human Factors group of a local company. She had a Bachelor's degree in Kinesethology. The other participant was a male then working as a software designer and teaching multi-media design. Both participants were classmates in a graduate course on HCI research methods. Both participants contributed to the data set. Neither had previous experience with results synthesis. The participants did not complete the results synthesis within the allotted time, but felt that given more time they would have completed the task.

The system worked well and was used largely as expected. As before, the participants were able to work together and apart and to move between the two without great effort. This pair of participants had a fairly sharp separation of activity early in the task. This was not explicitly coordinated, it "just happened." There was a noticeable level of effort to resynchronize when the participants needed to work together, but they were able to accomplish the transition and subsequently worked more closely together. The participants made and understood diectic references. The participants moved problems out of their initial groupings and into new categories of their own creation. They also used spatial layout. One interesting observation is that some of the conventions I was using in the system differ amongst various operating systems. In particular, some of the choices I had made – the way I deal with panning the workspace in the main view and group selecting –

were “opposite” of what the convention is for applications running on the MacOS™. This came to light because one of the participants was a “native” MacOS™ user.

No reproducible software defects were discovered in this study. A number of minor changes were identified, but were not implemented due to time constraints:

1. The key binding for creating freehand annotations is obscure. One of the participants wanted to create a freehand annotation, but could not determine what key combination would create a freehand annotation.
2. Problem reports are better presented as outlines rather than the tabbed dialogs. The tabbed window control prevented the user from seeing the content of the other sections, which would have been useful for referring to what was written in the other sections.
3. Have problem report content show up in the info area. As problem report creation is spread out over time, the users wanted to be able to refer to other problem reports spread throughout the workspace, without having to reposition the main view away from their current focus.
4. Modify the info area to deal with content larger than can be displayed in the currently allocate space. Some of the more verbose problem descriptions would not fit within the info area as currently configured. Being able to see the entire content of the card is useful. Assuming that the previous change (#3) was also made, problem report content would appear in the info area. This content is expected to be quite extensive, and as such could not be displayed usefully in the info area as currently implemented.

6.4.4 Study #7 - Two user

One participant was a male undergraduate student in Psychology, with previous traning in HCI. The other participant was a male graduate student in Environmental Design. Both participants were classmates in a graduate course on HCI research methods. Both participants were contributors to the data set. Neither had previous experience with results synthesis. The participants did not complete the task in the time allotted, but they indicated that they believed they could complete the task given more time.

The participants worked both as individuals and as a group, and were able to move between the two. The participants moved the cards out of their initial layout into new groupings of the participants' own discovery. This pair made more use of existing heuristic categories than participants in the other studies. They also made less use of empty space towards the bottom of the workspace. They tended to organize in place. Piles were used to remove clutter in the interface by tightly grouping cards that related to the same problem as had been observed in the previous studies.

A few of software defects were detected in this study, but have not been fixed. A number of additional minor changes to the interface were suggested but not made:

1. Ability to clone the card on the top of a pile without having to remove it from the pile. When piles are used to collect cards that relate to the "same problem," it may emerge that cards in one piles also seem to belong in another pile. Currently, the card would have to be removed from the pile to be cloned and then placed back in the pile. The heavyweight nature of this action sequence could prevent users from creating the richest possible representations.

2. Ability to find the clones of a given card. When cloning is extensive, it is often useful to be able to locate all the locations where a problem description appears. Often the users will lose track of whether or not a clone has been placed in another group. This mechanism would help prevent the creation of unnecessary clones. Also, being able to locate clones would aid in discovering relationships amongst seemingly disparate groups that might otherwise not be noticed until late in the process, or not at all.
3. Mechanisms for slaving or synchronizing views. When one user speaks to other users, referring to elements that are within the first user's main view, the other users must attempt to interpret what the first user is talking about. In the current system, the other users are faced with a decision of whether to try to figure out the situation without moving their main views, using the overview and info area only. The alternative is to move their main views, disrupting what they are doing, in order to better understand what the first user is talking about. A mechanism similar to Gutwin's (1997) teleport would provide a users with a lightweight way of seeing what the first user sees without losing their own context.

6.4.5 Study #8 - Three user

The primary goal of this study was to get a feel for the scalability of the system both in terms of system performance and user performance and comfort. Results synthesis is supposed to be carried out by a multi-disciplinary team that includes at least the inspectors of the interface. This means that in a real use I would expect there to be at least three users of the system engaging in the real-time results synthesis task.

One of the participants in this study (P3) was the experimenter. Another of the participants (P2) had participated in the single user studies and thus was familiar with both results synthesis and an earlier version of the system. The third participant (P1) had no previous experience with results synthesis. P1 was a male professor in Mechanical Engineering with no formal HCI training, though at that time he had been involved in building and evaluating an number of groupware systems. None of the participants were contributors to the data set, though P3 had reviewed the data set and had observed others performing results synthesis with it. The participants did not complete the task in the allotted time, though they indicated that they believed they could have completed the task given more time.

The participants worked both as individuals and as groups, though the coordination did not seem to be close – there was more working apart and more effort was expended in resynchronizing. Seldom were all three participants engaged simultaneously in a single task after the familiarization stage. The participants exploited the system features in much the same way as had been seen in the previous studies. Spatial layout was used extensively. The participants created their own groupings, moving the cards out of their original layout. Diectic references were made and understood, though the participants often needed clarification – verbally requesting that the person making the reference continue their demonstration or emphasize it until the other participants could figure out what was being referred to. A typical exchange might go:

P3: What do you think of this? <Clicks on card and jiggles it.>

P1: Where?

P3: This one here. <Jiggles card more demonstratively>

P1: Oh, yeah, that seems OK.

One software defect was encountered, but could not be subsequently reproduced. Two minor changes have been suggested as a result of this study but have not been implemented.

The first change is to provide drag scrolling. Often users would want to move a element to a new location that was just outside the area covered by the main view, or they would want to move the element in a particular direction without having a definite target, searching for the right location for the element as it was moved. The existing mechanisms for moving elements outside the main view – dropping to the overview or TA list – do not easily support these actions. They require that a target be defined and that the user remove their focus of attention from the main view. Drag scrolling allows a user to reposition both their view and an element at the same time. When the user drags an element beyond the boundary of the main view, the main view would be scrolled in the direction the user had dragged the element. The location of the element within the workspace is also updated. This is very convenient if the element is to be moved only a short distance, and the user intends to continue to work in the vicinity of the new location.

The second change is to provide some sort of “space warping” mechanism. When working in a fixed sized space, users often get into a jam because they have not anticipated the amount of space needed to construct their preferred layout of the elements in a grouping. Indeed, since they do not and cannot know what the elements of the grouping will be until it is complete, as membership in the groupings is an emergent property, they cannot

anticipate how much space will be needed for a particular grouping. This leads to frustrated attempts at spatial layout when the user runs into an edge of the workspace or starts encroaching on the space allocated to another grouping. This makes it difficult to express relationships through spatial layout in the desired way. This leads to one of two inefficiencies. Either time is spent on housekeeping operations at the time when the users notice there is inadequate space for the grouping, or time is spent later in extra effort to interpret and disambiguate groupings that have been shoehorned into a restrictive area and potentially intermingling with neighbouring groupings. A mechanism that would allow the users to give more space to a grouping would eliminate the need for these distracting activities.

6.5 Results and critique

In terms of achieving the goals of my research set forth in Section 1.4, this suite of studies proved successful. I have shown that the system I created and refined can be used to perform results synthesis. The system works well as proof-of-concept, both from a software reliability perspective as well as fitness for the task. The system described in this chapter represents a reasonable construction of how to support results synthesis in a real-time groupware system. I do not claim this is the best, or only way to support results synthesis. Rather, the design presented is a conservative, theory based one that employs current best practices (Shipman and Marshall, 1995; Marshall and Shipman, 1995; Gutwin, 1997) in the creation of real-time groupware supporting emergence.

I faced a number of challenges in performing this evaluation:

1. Evaluating for a diverse user audience. The system is designed to be used by participants with widely varying backgrounds. This presents a challenge in knowing how far to go in supporting something that may be particular to one group or another. This may vary from operating system conventions to preferred variations on the process or how information is presented.
2. I found little indication for radical redesigns of the interface. I am wary of this because I am not confident that the current interface is optimal. Perhaps with more rigorous data collection and more subjects, subsequent analysis would reveal evidence that would lead to more innovative interfaces and interaction methods. The design presented is a reasonable one, but my gut feeling is that more radical designs using alternate workspace visualization and awareness mechanisms (Gutwin, 1997) may be better suited to the task.
3. I found it hard to know when to heed the indications of the study when I thought doing so went against the theory on which the design is based, or were idiosyncratic to the particular participants. The use of piles to gather together problem descriptions that are part of “the same problem” is an example of a behaviour that I consider dangerous on theoretical grounds. By putting the cards into a pile, you are removing all but the top card from consideration. In the case of duplicates, this is not a problem, but if done prematurely, it may lead to over simplification of the data and the users may miss connections that they would otherwise have noticed if they had been forced to confront all of the data. In a related vein, one requested change was to allow problem descriptions to be place “inside” a problem report as a way of associating the report and the

descriptions as well as reducing visual clutter in the workspace and demarcating things that were “done” from those that still needed to be dealt with. While such functionality would not be a problem if used at the very end of the process, it may be damaging if used too earlier. Again, it removes elements from the workspace, removing them from consideration. This sort of feature promotes the appearance of certainty and does not afford the sort of equivocation and “try-it-and-see” attitude that is key to emergence. The danger is that the emergence phase degenerates into a sorting task.

In none of my studies did the participants complete the task set for them. However, I believe that the participants in the studies got far enough along to see that the process would complete given more time – that the participants’ suggestions that this was the case were not made solely to please the experimenter. In many of the studies the participants created some problem reports and in those cases they often completed one or more problem report. In all cases, significant progress was demonstrated – the users did not perform aimless operations, but proceeded in systematic manner and progressed in their plan to make sense of the data. Having said that, I believe that this is a part of results synthesis that deserves further study. In my studies I have not looked at the quality of the problem reports, or whether the problems identified in the reports are “good” ones.

6.6 Conclusion

In this chapter I have described the formative evaluation of the interface I presented in Chapter 5. The evaluation was tied to progressive interface refinement, where the results of a study were used to modify the interface before the next study. Eight studies were conducted, three in a single user scenario and five in a multi-user scenario. Many software

defects in the system were found and fixed. A number of significant changes to the interface were made, in particular the addition of the info area and notes. The main result of the evaluation is that the PReSS system is a reasonable way of supporting results synthesis and that representative users can perform results synthesis using the system. This meets the goals for my research set out in Section 1.4.

Chapter 7: Conclusion

7.1 Introduction

In this thesis I characterized results synthesis and showed how it might be supported in both paper based and distributed groupware environments. The research was motivated by my desire to improve the effectiveness of Heuristic Evaluation. Other researchers have pointed out that the output of results synthesis is key to the effectiveness of Heuristic Evaluation, yet there had been nothing written about this process before now. Furthermore, I believed that environments could be created that would support the results synthesis process, making it easy for the participants to achieve the desired qualities in the output. While performing the research, my perspective has been that of a researcher, designer, and practitioner. Consequently, I have investigated results synthesis with the goal of making Heuristic Evaluation effective in practical application.

In Section 7.2 I revisit the research hypothesis and goals set out in Chapter 1 and summarize how my research has met these goals. In Section 7.3 I summarize the main contributions of my research as well as well as the incremental improvements made as a result of my research. Additional research questions that arose in the course of this research, but were beyond the scope of my thesis are treated in Section 7.4.

7.2 Research goals and summary

My research hypothesis was that results synthesis in Heuristic Evaluation is a definable and describable process, that constraints on the process may be identified, and that environments may be created that support this process. This led directly to the goals of my research:

1. Define and describe results synthesis.
2. Identify requirements for supporting results synthesis.
3. Construct and evaluate a prototype system for supporting results synthesis.

I have met these goals through the course of my research as detailed below.

Define and describe results synthesis. In Chapter 2, I presented my definition of results synthesis within the context of Heuristic Evaluation:

> Results synthesis is the process of transforming the entire collection of raw problem descriptions into a coherent, complete, and concise statement of the problems in the evaluated interface along as well as recommended actions to address the problems identified.

I then described results synthesis as a participatory practice in Chapter 3. I reported on my observational studies of groups performing results synthesis in paper based environments. From these observations I synthesized a scenario describing a "typical" results synthesis session.

Identify requirements for supporting results synthesis. In Chapter 4 I presented a set of requirements for the support of results synthesis. These requirements are based on the literature of Heuristic Evaluation, participatory practice, and emergence. Each requirement is accompanied with theoretical justification from the literature and is in accordance with my observations of how the process unfolded in a paper-based environment. In setting forth these requirements I have met the second goal of my research.

Construct and evaluate a prototype system for supporting results synthesis. To meet the first part of this goal I designed and implemented PReSS, a prototype groupware system for supporting results synthesis (Chapter 5). The system is based on the metaphor of the environment used in paper-based results synthesis. I leveraged the advantages provided by a computational medium to go beyond a strict interpretation of the metaphor, while mitigating the disadvantages of working on a small display. To fulfill the second part of my research goal I performed an iterative formative evaluation of the prototype system (Chapter 6). The goal of this evaluation was to refine the interface and validate its ability to support users in carrying out results synthesis. I ran eight trials in differing conditions that resulted in a number of changes to the interface and led me to conclude that the system does in fact provide reasonable support for results synthesis by distributed groups.

7.3 Research contributions

There are two major research contributions in this thesis. The first is the definition of results synthesis as a process and as a research area. Previously, this activity had been described simply as "aggregation," which suggests much less complexity than is actually

present. The question of how a collection of raw problem reports gets turned into a series of polished final problem reports has not been addressed previously, even though the quality of those final problem reports is crucial to the effectiveness of Heuristic Evaluation in carrying out its stated purpose (Jeffries, 1994; Sawyer, Flanders, and Wixon, 1996) – the improvement of a user interface.

The second major contribution is identification of emergence as central to results synthesis. There is no simple, linear, predictable path from the raw problem descriptions to the final problem reports. This distance can only be spanned by allowing those performing results synthesis to explore the space without constraint, finding a route that fits their particular situation. This has been verified in the studies I conducted, both in paper-based and distributed groupware environments.

This research has also produced a number of less important but still significant advancements:

- I have described a participatory approach to results synthesis and provided a theoretical justification for it.
- I have put forward a list of requirements for supporting results synthesis in any environment.
- I have created a prototype distributed groupware system for results synthesis and shown that it does provide reasonable support for the activity.
- I have supported Gutwin's (1997) findings on workspace awareness by following his guidelines in designing the system, which resulted in a system that was usable by distributed groups in carrying out a task much different from the ones used in his research.

7.4 Further research suggested

On the road to meeting my research goals, I came across many interesting questions that I could not address because they were beyond the scope of my research. In this section I will put forth what I take to be the most important or interesting of these questions. I have divided these questions into two groups: those dealing with results synthesis independent

of the environment used, and those dealing specifically with system support for results synthesis.

7.4.1 Results synthesis process

Further research on the results synthesis process suggested by my research falls into two major themes: mapping the space, and understanding the data.

Mapping the process space. My research has, of necessity, been tightly focussed. There are a number of dimensions of variability in how the results synthesis process is configured that deserve to be explored. I have argued that a participatory approach to results synthesis will bring the greatest benefit to the organization carrying out the evaluation. The studies I performed and the system I design assumed this approach to results synthesis. Other approaches need to be investigated so that we can understand the differences between having one person perform results synthesis and having a group perform the same task. As well, the participatory approach deserves further study, as the groups used in many of my studies were not as participatory as would be seen in real development contexts, due to my constraints on procuring study participants.

Another dimension of variability in the process space is development context. Most of my studies had the participants using raw problem descriptions derived from a simple, obviously deficient interface described through a screen snapshot and an accompanying written description of its behaviour (Appendix A). More research is needed to look at results synthesis in a wider variety of development contexts: designs that are both earlier and later in the development process; more complex interfaces; interfaces with more subtle or ambiguous problems; and interfaces that generate many more raw problem descriptions.

The final dimension of variability is the organization of the process itself. In my research, I have described a very high level process for carrying out results synthesis. The participants in the studies found the high level description sufficient to allow them to successfully undertake this process. Any attempt to perform it will have to have the four stages I identified as constituents of this process. However, alternate ways of performing the stages, and alternate ways of interleaving the stages need to be investigated.

Understanding the data. At present we do not have a good understanding of the nature and impact of the data involved in results synthesis, both the input data (raw problem descriptions) and the output data (final problem reports). With respect to the input data, more research is needed to understand how variations in what is recorded and how it is recorded impact results synthesis. In many of my studies, the raw problem descriptions were very concise, often consisting of a single phrase or sentence. At other times, the problem descriptions were very verbose. How this impacts results synthesis needs to be investigated.

One recurring theme in the studies was the notion of components or parts of the interface as an important way of organizing or talking about the problems during some portion of the emergence stage. The role of the interface structure in organizing problems needs to be researched. While I do not believe that the interface structure leads directly to the final problem reports, it does have some role to play. This role needs to be further investigated, so that it may be effectively exploited and so that raw problem descriptions record this information, if it is important.

If we are to judge the quality of a results synthesis process without actually embedding it within a development process, then we must develop a means of judging the quality of final problem reports. Further research into the nature of effective final problem reports is needed.

7.4.2 System support for results synthesis

I have discerned three areas of further research on system support for results synthesis: leveraging computer representation, mitigating the small display, and group support.

Leveraging computer representation. One of the promises of using a computational environment for results synthesis is that some actions are much easier than they would be in a paper based environment. Examples from PReSS (Chapter 5) include cloning of cards, moving and deleting text annotations, and the automation of the initial layout of cards. The possibility of automating other parts of results synthesis should be investigated. However, the computer scientist's desire to automate everything should be tempered by the need to show that the automation at least maintains the quality of the output. Given the

large role of judgement, context, and experience in results synthesis, I believe increased automation is most likely to come in the form of performance support, rather than replacing the human element.

The prototype system presented in Chapter 5 does not represent the complete functionality expected in the system. An example of one feature not currently implemented was "properties." I intended this feature to provide visual cues (part of requirement 17 in Chapter 3). By "visual cues" I mean changing, for instance, the colour and shape of an alias as another means of indicating relationships between problem descriptions or other elements in the workspace. For example, all of the cards representing problem descriptions ascribed to the same heuristic could be given the same colour, and all the cards authored by one individual could be given a distinctive shape to allow the users to determine who wrote what. The best use of these limited channels of signalling relationships between the elements in the workspace need to investigated further. In particular, what information is best encoded in shape and colour and whether this should be system assigned or user configurable needs to examined.

The prototype, I believe, was quite simple in terms of the functionality it offered. Nonetheless, it had a noticeable learning curve. The addition of increased functionality raises the issues of how users are going to access this increased functionality, and what sort of training is appropriate so that users will exploit all the system is capable of doing.

Using a system to capture raw problem descriptions as mentioned in Chapter 5 opens up the possibility of *rich problem capture*. Rich problem capture is the ability to record more than just text, including gesture, annotation, audio commentary, and images or animations of the interface under evaluation. The question of how to present rich problem descriptions in a system supporting results synthesis is one that needs further consideration. All the additional data is going to be competing for already scare display space.

Mitigating the small display. Systems supporting results synthesis have displays that are at least an order of magnitude smaller than the area used in paper based environments. Thus, one of the most important design decisions in creating such systems is how the system can approach the facility with which people naturally engage the workspace in paper-

based environments. The design presented in Chapter 5 is adequate for the task, as shown in Chapter 6, but it is conventional and conservative (Gutwin, 1997). Alternative workspace visualization techniques, such as zoomable windows and fisheye visualizations need to be investigated (Gutwin, 1997). When considering these unusual techniques, in addition to their theoretical fitness to the task, we must also investigate how they impact the usability of the system by novice users in accordance with requirement 26 from Chapter 3.

Non-visual techniques for dealing with the workspace also need further investigation. A textual search feature was requested by a few participants in the studies from Chapter 6. Other ideas worth further investigation, in my opinion, include the presentation of some system-computed notion of similarity to aid users in finding related problem descriptions. Also, the metaphor of cards on a wall is not necessarily the best basis for results synthesis on a small display, and other ways of presenting the data, consistent with the requirements set out in Chapter 3, should be investigated.

Group support. The prototype system described in this thesis does support distributed groups in performing results synthesis. However, I saw an increase in collaborative overhead when moving from two participants to three that was disproportionate to what I would expect from a similar increase in a paper based environment. To me, this means that the prototype system does not represent all that can be done with respect to workspace awareness, though it contains the core of what is recommended by Gutwin (1997). Additional support needs to be investigated for groups of at least three users.

7.5 Conclusion

In this thesis I have defined results synthesis as a process and as an area of research, and operationalized those definitions. The goals of my research were: to define (Chapter 2) and describe (Chapter 3) the process of results synthesis; to understand how this process could be supported (Chapter 4); to create a system based on that understanding (Chapter 5); and to show that the system can be used to successfully perform results synthesis (Chapter 6). I met each of these goals in the course of my research, as described in Section 7.2. The major contributions, as detailed in Section 7.3, have been the definition of results synthesis as a process and as key element in the effectiveness of Heuristic Evaluation, as well as the

key role of emergence in the results synthesis process. Finally, I have identified a research agenda for results synthesis in Section 7.4 that moves us towards a level of understanding where we can say how to perform results synthesis in order to maximize the effectiveness of Heuristic Evaluation.

References

Bias, R.G. (1994). Pluralistic Walkthroughs. In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*. Wiley.

Bias, R.G. and Mayhew, D.J. (1994) *Cost-Justifying Usability.* Academic Press.

Boehm, B.W. (1988). “A Spiral Model of Software Development and Enhancement,” Computer, 24(May 1988), 61-72.

Clark, H. (1996) *Using Language*. University of Cambridge Press.

Desurvire, H. (1994). Faster, Cheaper!! Are Usability Inspection Methods as Effective as Empirical Testing? In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*. New York NY: Wiley.

Dumas, J.S., and Redish, J.C. (1993). *A Practical Guide to Usability Testing*. Ablex Publishing.

Edmonds, E.A., Moran, T.P., and Do, E.Y. (1998). Interactive system support for the emergence of concepts and ideas. *SIGCHI Bulletin*. 30(1), 24-25.

Englebart, D. (1968). Augmenting Intellect. In *Proceedings of the Summer Joint IJCC Conference*.

Greeberg, S. (1996a) . Teaching Human Computer Interaction to Programmers. *interactions*. 3(4), pp. 62-76.

Greenberg, S. (1996b). Course notes from “Graphical User interfaces: Design and Usability”. Continuing Education, University of Calgary. (http://www.cpsc.ucalgary.ca/~saul/609.05/index.html).

Grudin, J. (1988). Why CSCW applications fail. In *Proccedings of the CSCW ’88 Conference*. ACM. pp. 85-93.

Gunn, C. (1995). An Example of Formal Usability Inspections in Practice at Hewlett-Packard Company. In CHI ’95 Conference Companion. pp. 103-104.

Gutwin, C. (1997). *Workspace Awareness in Real-Time Distributed Groupware*. Ph.D. Thesis, University of Calgary.

Humphreys, W.S. (1995). *A Discipline for Software Engineering*. Addison-Wesley.

Humphreys, W.S. (1989). *Managing the Software Process*. Addison-Wesley.

Jeffries, R. (1994). Usability Problem Reports: Helping Evaluators Communicate Effectively with Developers. In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*. New York NY: Wiley.

Jeffries, R., Miller, J.R., Wharton, C., and Uyeda, K.M. (1991). User interface evaluation in the real world: A comparison of four techniques. In *Proceedings of the ACM CHI '91 Conference*. pp. 119-124.

Kahn, M.K. and Prail, A. (1994). Formal Usability Inspections. In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*. New York NY: Wiley

Karat, C. (1994). A Comparison of User Interface Evaluation Methods. In Nielsen, J. and Mack, R.L. (Eds.), *Usability Inspection Methods*. John Wiley and Sons. New York, NY.

Mack, R.L., and Montaniz, F. (1994). Observing, Predicting, and Analyzing Usability Problems. In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*. New York NY: Wiley.

Marshall, C.C., and Rogers, R.A. (1992). Two years before the Mist: Experiences with Aquanet. *ACM ECHT Conference*. pp. 53-62.

Marshall, C.C., and Shipman, F.M. (1995). Spatial Hypertext: Designing for change. *Communications of the ACM*.. 38(8), 88-97.

Marshall, C.C. and Shipman, F.M. (1993). Searching for the Missing Link: Discovering implicit structure in spatial hypertext. In *Hypertext '93 Proceedings*,. pp. 217-230.

Marshall, C.C., Shipman, F.M., and Coombs, J.H. (1994). VIKI: Spatial Hypertext Supporting Emergent Structure. In *Proceedings of ECHT '94*. pp. 13-23.

McCleverty, A. (1998). Personal communication.

Molich, R. and Nielsen, J. (1990). Improving a human-computer dialogue: What designers know about traditional interface design. *Communications of the ACM*. 33(3), 338-348.

Monty, M. (1990). *Issues for Supporting Notetaking and Note Using in the Computer Environment*. Ph.D. Thesis, University of California, San Diego.

Moran, T.P., Chiu, P., and van Melle, W. (1997). Pen based interaction techniques for Organizing material on an electronic whiteboard. In *Proceedings of the ACM UIST '97 Conference*. pp. 45-54.

Muller, M.J. (1998). Ethnocritical Heuristics for Reflecting on Work with Users and Other Interested Parties. In Kyng and Mattiassen (Eds.), *Computers in Context*. MIT Press.

Muller, M.J. (1997). Personnal communication.

Muller, M.J. (1993). PICTIVE: Democratizing the Dynamics of the Design Session. In D. Schuler and A. Namioka (Eds.), *Participatory Design*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Muller, M.J., Matheson, L. Page, C., and Gallup, R. (1998). Participatory Heuristic Evaluation. *interactions*, 5(5), 13-18.

Muller, M.J., McClard, A., Bell, B., Dooley, S., Meisky, L., Meskill, J.A., Sparks, R., and Tellam, D. (1995). Validating an extension to participatory heuristic evaluation: Quality of work and quality of work life. In *CHI ’95 Conference Companion*. (Denver, 1995), 115-116.

Muller, M.J., Dayton, T., and Root, R. W. (1993). Comparing studies that compare usability methods: An unsuccessful search for stable criteria. *INTERCHI ’93 Adjunct Proceedings*, pp. 185-186.

Munemori, J. and Nagasawa, Y. (1996). GUNGEN: groupware for a new idea generation support system. *Information and Software Technology.* 38(1996), 213-220.

Nielsen, J. (1995a). Keynote to the IFIP INTERACT ’95 Conference.

Nielsen, J. (1995b). Scenarios in Discount Usability Engineering. In J. Carroll (Ed.), *Scenario-Based Design*. John Wiley & Sons.

Nielsen, J. (1994a). Enhancing the explanatory power of usability heuristics. *In Proceedings of the ACM CHI ’94 Conference*. (Boston, 1994), 152-158.

Nielsen, J. (1994b). Heuristic Evaluation. In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*. Wiley.

Nielsen, J. (1993). *Usability Engineering*. Academic Press.

Nielsen, J. (1992). Finding usability problems through heuristic evaluation. In *Proceedings of the ACM CHI ’92 Conference*. (Monterey, 1992), 84-90.

Nielsen, J. and Mack, R.L. (1994). *Usability Inspection Methods*. Wiley.

Nielsen, J. and Molich, R. (1990). Heuristic evaluation of user interfaces. *In Proceedings of the ACM CHI ’90 Conference*. (Seattle, 1990), 249-256.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. and Carey, T. (1994). *Human-Computer Interaction*. Addison-Wesley.

Roseman, M. and Greenberg, S. (1996). Building Real-Time Grouware with GroupKit, a Groupware Toolkit. *Transactions on Computer-Human Interaction*. 3(1), pp. 66-106.

Rubin, J. (1994). *Handbook of Usability Testing*. John Wiley & Sons.

Rudd, J., Stern, K. And Isensee, S. (1996). Low vs. High Fidelity Prototyping Debate. *interactions*. 3(1), 76-85.

Saarinen, T., and Saakjarvi, M. (1989). The missing concepts of user participation: An empirical assessment of user participation and information system success. In *Proceedings of the 12th IRIS* (Information Systems Research in Scandinavia). (Aarhaus, 1989), 533-553.

Sawyer, P., Flanders, A., and Wixon, D. (1996). Making A Difference - The Impact of Inspections. In *Proceedings of the ACM CHI '96 Conference*. pp. 376-382.

Schuler, D. and Namioka, A. (1993). *Participatory Design*. Erlbaum.

Shipman, F.M., and Marshall, C.C. (1994). *Formality Considered Harmful: Experiences, Emerging Themes, and Directions*. Xerox PARC Technical Report ISTL-CSA-94-08-02.

Wickens, C. D. (1992). *Engineering Psychology and Human Performance, 2nd Ed*. HarperCollins.

Winograd, T. and Flores, F. (1986). *Understanding Computers and Cognition*. Addison-Wesley.

Wharton, C. (1994) Cognitive Walkthoughs: A practioners guide. In J. Nielsen and R.L. Mack (Eds.), *Usability Inspection Methods*. Wiley.

Appendix A: Example Interface Description

The interface description on the following page comes from Nielsen's *Usability Engineering* (1993, p. 234-5). This is the interface that was inspected to obtain the raw problem descriptions used in the first observational study (Section 3.6) and all of the PReSS evaluation studies (Chapter 6).

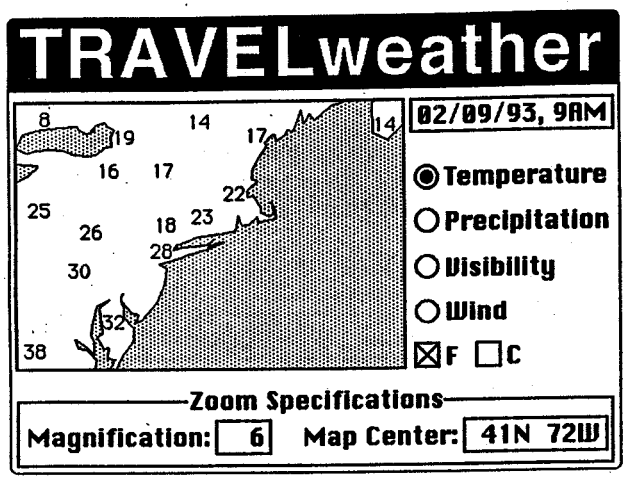


Figure 24 *Screen design for a hypothetical system to provide weather information and forecasts to travellers.*

Exercise 8: Heuristic Evaluation of a Paper Mock-Up

Figure 24 shows a design for a system to provide weather information to travelers. *TRAVELweather* (a non-existing system) can provide information about the weather at 3AM, 9AM, 3PM, and 9PM for the current day as well as the two next days, using reported readings for past weather and forecasts to predict future weather. The interface is designed for use on a graphical personal computer with a mouse, and will appear in a separate window on the screen.

The user operates the interface by typing the desired time into the box in the upper right part of the screen. If the user types a date other than today or the next two days, or if the user types a time other than the four times for which information is available, the system will show an alert dialog box with the following error message: "`Weather Data Not Available`." The only button in the error message box is an "OK" button. Clicking the OK button will make the dialog box go away and will reset the date and time specification to the previous value.

The user changes the map display by editing the boxes for zoom magnification and for the center of the map. The system ensures that only integer numbers can be typed in the map magnification box by simply beeping every time the user presses a non-number key in that box. If the user types anything in the map center box other than a valid set of coordinates (an integer from 0 to 90 followed by the letter N or S followed by an integer from 0 to 179 followed by the letter W or E), the system will show an alert dialog box with the following error message: "`Unknown Map Coordinates`." The only button in the error message box is an "OK" button. Clicking the OK button will make the dialog box go away and will reset the coordinates to their previous value.

With respect to all three input boxes, the user's changes take effect as soon as the user clicks the mouse outside a box after having edited it.

Perform a heuristic evaluation of this interface. Remember to evaluate the entire interface, including both the figure and the text describing what happens as a result of various user actions. The result of the heuristic evaluation should be a list of the usability problems in the interface with reference to some established usability principle violated by that aspect of the interface. It is not sufficient just to say that you do not like something; explain *why* you do not like it with reference to the heuristics in Chapter 5 or other usability results. Try to be as specific as possible and list each usability problem separately. For example, if there are three things wrong with a certain dialogue element, list all three with reference to the various usability principles that explain why that particular aspect of the interface element is a usability problem. There are two main reasons to note each problem separately: First, there is a risk of repeating some problematic aspect of the dialogue element, even if it were to be completely replaced with a new design, unless one is aware of all its problems. Second, it may not be possible to fix all usability problems in an interface element or to replace it with a new design, but it could still be possible to fix *some* of the problems.

Appendix B: Example Instructions to Inspectors

Heuristic Evaluation Exercise

The main point of this is to pay attention to how people organize the data that results from doing heuristic evaluation. To that end:

1. Find 10 - 20 user interface problems with the example interface. For this exercise we are using the TRAVELweather example from Nielsen's Usability Engineering. I have distributed copies of the exercise along with these instructions.

2. For each problem, on a 4"x6” index card, use a marker to write a sentence or short description of the problem and label it with one of the following heuristics:

> Visibility of system status
> Match between system and the real world
> User control and freedom
> Consistency and standards
> Error prevention
> Recognition rather than recall
> Flexibility and efficiency of use
> Aesthetic and minimalist design
> Help users recognize, diagnose, and recover from errors
> Help and documentation
>
> For a more detail description of the heuristics, see <http://www.useit.com/papers/heuristic/heuristic_list.html>.

3. We'll meet at 1pm in MS 623 unless otherwise notified. I imagine it will take at least one hour, but not more than two.

Thanks,

Donald Cox

Appendix C: Raw Problem Descriptions from Scenario

The following raw problem descriptions are the ones used in the scenario presented in Chapter 3. They are a subset of those generated for the first observation study, which is the same as the DS1 data set (Table 6.1) used in Chapter 6.

[] F [] C not clear what they are going to mean for, in precip, visib, etc... H: Consistency and Standards	Consistency & Standards Not clear what F & C mean when something other than temp is selected
Temperature/Precip/Visibility/Wind not visible all at once. User is usually trying to get a picture of the general weather H: Consistency and standards H: Flexibility & Efficiency of use	User control & freedom • no history or "bookmark" feature
No accelerators for date - e.g. flip between times/dates (arrow keys) H: flexibility and efficiency of use	User control & freedom (8) No way to return to prev. coords/settings without remembering/retyping them
Flex & effic of use (7) • no way to scan a set of 'standard' places of personal interest over time	No undo/redo capability for date H: User Control and freedom
Consistency and Standards user must click outside entry box for changes to take effect. Design: add enter key	Match (2) + aesth (8) • graphics confusing as to what is land & water • also 'busy' (water dots)
User must click outside a box to get update - may mistake current info for requested info. H: Visibility of status	Help & documentation There isn't any

Help & doc (10) None!	Flexibility & Efficiency • allow different date/time formats
Aesthetic and Minimalist design Switching between °F and °C would be very infrequent Design: tuck this option away somewhere	Concis. (4) check button meanings => radio button!
Consistency & Standards F and C are exclusive choices, assumedly, so they should have radio buttons instead of check boxes	Error Prevention (5) - date and time format not clear for data entry, especially day/month :: error prone
Shouldn't type/display date as xx/xx/xx - users don't know with is month/day (H) Consistency and Standards	Error Prevention (5) also User should not be able to type in data outside the range
Shouldn't type date - too easy to make mistakes H: Error prevention	Match (2) "Magnification meaning not clear in this context
Match between sys. & real world zoom spec. is arbitrary Design: slider or distance	Match (2) + (5) Value for mag mysterious
Match... (2) zoom spec: not in user's language	Magnification numbers mean nothing (what is the scale?) H: Consistency and Standards

Match btw system & real world • Display should have place names associated with data points.	User types in coordinates - doesn't know where he wants (e.g. what are lat/lon. of Boston H: Recognition Rather than recall
Match btw system & real world Map center should be in something other than lat./lon. Design: use direct manipulation	Match (2) coordinates for map center unfamiliar to most (all?) people
User can't "browse" a region, must enter new coordinates (e.g. shift landscape) H: Flexibility and efficiency of use	Const. & standards (4) Most end user maps navigated by panning. This one isn't.
User changes map location using lang/lat. - USER thinks in terms of destination or regions H: Match between system and real world	Recognition rather than Recall • constraints on input are not available
Error Prevention input of map center and date/time should constrain user to valid inputs.	Consistency and Standards user must click outside entry box for changes to take effect Design: add enter key
Rec. over recall (6) + 2 + 1 • clicking outside of box to activate input must be learnt/memorized • means its moded (1)	Aesthetic and Minimalist Design How important is visibility
Flexibility & Efficiency • only one data type displayable at a time	Aesthetic & Minimalist Design large amount of space wasted on title

Zoom Specification title + box - Unnecessary Information H: Aesthetic & Minimalist design	Visib. (1) + 3 When zoom into new location. Maybe (say) all white with no landmarks – context is lost
Visibility of system status • Distinction between actual and forecast conditions not made apparent in interface	Consistency & Standards What units will precic. visibility & wind be displayed in?
Use of OK button on Date Error Dialog - help missing H: Help and documentation	User Control and Freedom • Not explanation of why weather data is not available
Error Recovery ... (9) • weather data not available doesn't say how to fix the problem, or what the problem is (e.g. "wrong date entered")	OK on the "Unknown Map Coord" box - need help Help & Documentation
Error recovery (9) 'Unknown map coords' not helpful	Help users deal with errors "Unknown map coordinates" doesn't indicate what known map coordinates are.
Consis & standards (4) • several diff't ways of showing errors - audible beeps - dial. box	Only integer numbers error results in beep Help users recognize, diagnose, & recover from errors
Error recovery (9) "Beep" on non-integer in mag box mysterious	User Control and Freedom • system resets date/time without indicating the specific part that is in error

Visibility (1) + (9) Dialog boxes may cover person's input : can't see what they had typed nor why it's wrong; because it is cleared by selecting OK, can never see it!	On Error dialogs - resets the system to previous values, not allowing the user to edit mistaken entries H: User control and freedom
User control and Freedom • OK button on error dialog when clearly things aren't OK	User gets wrong date - Error message doesn't offer recovery - weather data should be "forecast" H: Help users recog. & diagnose and recovery from errors

Appendix D: Final Problem Reports from Scenario

I have created a complete set of final problem reports for the data set used in the scenario (Chapter 3). These problem reports are my own invention, based on my insight from performing results synthesis with the DS1 data set (Table 6.1), a super-set of the one used in the Chapter 3 scenario. The groupings in the scenario, and hence the problem reports, did not occur in any of the studies, but reflect my own views.

D.1 Use Geographic Names

Problem: Most users of the interface will not be used to dealing with the world in terms of latitude and longitude. Rather they speak about the world in terms of geographic names of varying specificities. A user will be interested in the weather in the Maritimes, or southern Alberta, or Flin Flon. Most users will not have the resources necessary to convert these goals into the necessary latitude and longitude specifications. This will present an insurmountable barrier to using the system for many potential users. Those that do use the system are not likely to use it to its fullest extent because of the difficult of translating their desires into the appropriate actions. Also the lack of place names in the interface can make it very hard to understand what is currently being shown. Given the lack of geographic or political features shown in the display it is unlikely the users will be able to figure out what part of the world is being displayed and at what scale. To do so would force them to use a translation mechanism external to the interface to convert the latitude and longitude into a more familiar and sensible name. Further, forecasts are not given for a particular intersection of latitude and longitude in our experience, but rather for a named region – Prince Rupert, Labrador, or the prairies.

Severity: This problem will affect both novice and experienced users in most tasks they would perform. The need to use latitude and longitude will be very intimidating to many novice users, some of whom may not even understand the terms. While phenomenally motived users may be able to learn to use latitude and longitude, it is highly unlikely that they will be able to easily convert between that and the more commonly used names. Many are likely to rely on “cheat sheets” and not use the interface for getting information beyond a small set frequently used locations, perhaps only one. This constitutes a serious competitive disadvantage in our opinion.

Recommended Action: Latitude and longitude provide a good mechanism for moving to random locations throughout the world or in regions that have no recognizable features or conventional names such as deserts or oceans. However, this is outweighed by the difficulty most users would have in translating between level at which they naturally express their intent and the level at which the interface is controlled. Therefore, control of the interface should be done primarily in terms of familiar geographic names. Further, rather than having type in names, which is error prone, the user should be presented with a means of selecting from among the valid choices. One mechanism would be a hierarchical list or browser. Another might be a pie menu scheme based on geographic proximity. By having the map center a named location, the need to have place names in the display area itself is reduced, given that the display is already going to be crowded with weather information.

D.2 Options

Problem: Putting infrequently used controls on the main display wastes precious display space and potentially adds to user confusion. When dealing with the weather, the user will have a set preference in how he or she wants the data displayed – in this instance, whether the temperature is to be displayed in degrees Celsius

or degrees Fahrenheit. Once the choice has been made, the user will not likely want to change it, and if he or she does change it, then it will be very, very infrequent. Putting such infrequently used functionality as the Fahrenheit/Celsius control on the main screen wastes space that could be better used displaying data or controls that the user is going to be using more frequently. Further, since the controls are on the main screen, the user expects them to have an effect at any time – thus bringing into doubt what F and C might mean, as Fahrenheit and Celsius don't mean anything when precipitation is being display. Further confusing the issue is that these two exclusive choices have check-box controls, indicating that it is possible to have both active at once (which at least makes sense, but would overload a crowded display space) or to have neither active at once (which makes no sense when viewing temperatures). Cramming everything into the main screen makes these options more terse, and hence less understandable, than they need to be.

Severity: Once the users figure out what the controls mean, they are not likely to be impeded by them. However, there will be significant consternation in figuring out what they mean initially. Perhaps the best case scenario is that they will be ignored entirely. The wasted space issue is much harder to describe, but more space given to the display of the weather data could have a significant positive impact on the usability and usefulness of TRAVELweather.

Recommended Action: Create a separate dialog for controlling the various display option such as which temperature scale (Celsius or Fahrenheit) is used, as well as input/output options such as date and time formats. Some mechanism for invoking this dialog will also have to be designed, though we expect these things to be used so infrequently that it may be acceptable to put them in a separate configuration file that would be edited with another program, though this would only be a last resort. The important thing is that the unimportant controls be hidden during normal operation, yet remain accessible for the occasion when they are desired.

D.3 Error Handling

Problem: Errors are not handled in a fashion that helps the users understand what went wrong and correct their behaviour in the future. First of all, the beeping in the magnification input field, in the absence of any other feedback or explanation, does not indicate what the user should be doing – what the computer is expecting. The error messages say that something is wrong, but leave the user guessing as to the exact inadequacy of what they provided as input and this is compounded by the fact that the system erases what they had written, leaving them to guess what they wrote and how to come up with something different.

Severity: If the current input methods are not modified as recommended in the rest of this report, this is a problem that will effect all classes of users in all their tasks. It will be a major source of frustration, especially to novice and infrequent users. However, if the other changes recommended are made, then there will not be any need for error messages.

Recommended Action: The first recommendation is to eliminate the need for error handling, which would be accomplished by following the recommended course of action found in the other problem reports. If this cannot be accomplished, a number of things may be done to improve the error handling. First of all, the error messages should say what the legal input looks like and how what the used entered diverges from the expected form. If the system is able to narrow down what the error is, then it should indicate in a non-judgemental way where the system has trouble interpreting the user's input. The system should not erase the user's erroneous input, but rather hilight the part that is in error so the user knows exactly what needs to be fixed.

D.4 Display

Problem: Users want the display of weather data to be compact and comprehensive. A person's conception of "weather" is based on more than just one measurement. The current design only displays one measure-

ment at a time, forcing the user to remember all the individual datums and integrate these in his or her head to form a meaningful picture of what the weather is like at a location. Conventional displays such as weather maps on TV broadcasts and in newspapers typically show all the weather variables combined.

Severity: While learning the interface, this will not be much of a problem, but as the user attempts to use the system for more than curiosity – as a planning tool and a means of understanding what is going on – this deficiency will prevent the users from using the system as they desire.

Recommended Action: Use conventional symbols to display multiple weather variables simultaneously.

Problem: The users have trouble understanding where the display is located. The lack of names and geo-political boundaries makes it very hard for the users to figure out where in the world the display is centered. The only way, really, to do this is to recognize a bit of coastline, but that requires the user to recognize the difference between land and water, which is not immediately obvious from the interface.

Severity: This problem will affect many users regardless of their experience level. The majority of users without good geographic survey knowledge will never be confident of where they are looking, and all users will have to double check the latitude and longitude with some external resource when looking at one of the many areas that does not have distinct coastlines or large bodies of water.

Recommended action: Provide at least the name of the locale at the center of the display. Ideally, the names of all the forecast points should be shown, but if there is not enough display space, a facility like Balloon Help™ should be included so that the users can discover what the other places are.

Problem: Non-functional items consume much valuable screen space. The larger the percentage of window area that can be used to display data, the more usable and useful the program will be. A significant proportion of the current design is dedicated to non-functional items such as the program title and control grouping titles and boxes.

Severity: Compared to many of the other problems discussed in this report, this wasted space does not have an easy to discern impact on the users. However, reducing the amount of wasted space will improve usability and usefulness, if only because it will allow more comprehensive and verbose data displays as have been recommended in other problem reports.

Recommended action: Reduce or eliminate the window title. Branding can be achieve in other ways that are just as effective and do not have an adverse impact on usability. Further, if the other changes recommended in the other problem reports are implement, the controls will fit within the space now used by the title, allowing for a substantial increase in the area dedicated to weather data display.

D.5 Make Choices Visible

Problem: The user should not have to guess at input formats and values when there are only a small number of legal choices. In the existing TRAVELweather design, there are two places where the used has to guess. One is the Magnification input. The user must guess what the allowable values are, and what a particular value will mean in terms of what they will be able to see. The date and time input field only allows twelve valid inputs, and the user is forced to guess what these values are, though no clue exists in the interface as to what they might be. Further, the allowable formats are not communicated in any way by the interface. An additional problem is that the system does not know when the user has completed his or her input, and only processes the input when the user clicks outside the modified input box. The users will not develop the correct mental model for this, and may become confused because they will not know if the data being displayed is from their last setting or the current one.

Severity: The date part of this problem is the most important, as it is one of the most important pieces of functionality in the interface and will be used in almost all tasks. While it is possible that experienced, frequent users of the system could learn to used the date field, there are likely to still be many annoying slips. The problem with the magnification field is also significant and is likely to occur no matter what the experience level. It is unlikely that the users will develop an accurate mapping between the numeric values and what is going to be displayed. The problem of notifying the system of new input is very serious as it will affect all classes of users and they will not figure it out without being told. This will cause many misinterpretations of the data, and subsequent loss of trust in the system.

Recommended Action: Allow the user to chose a valid value. Rather than having to guess, it would be easy for the user to chose amongst the three valid dates and four valid times. This can be done through the keyboard and/or the mouse depending on what widgets are used. The magnification value should also cycle through the allowable choices, assuming there are only a few meaningful choices. Or, a slider-type control could be used, dispensing with the numbers altogether. By adopting these input techniques, the system automatically knows when the user has changed input values and will be able to update the display so that the information shown is always in accord with the input values displayed.

D.6 Moving in Space and Time

Problem: Users want to express themselves in their own terms, not some overly accurate, arcane system. The number of users who understand latitude and longitude will be minuscule. Of those, few will have the resources and the desire to translate between the level at which they form their intentions (i.e. What is the weather in Toronto) and the level at which the interface allows them to express their intentions (input latitude and longitude of Toronto). Latitude and longitude are also over-specific, as individual forecasts or weather readings are not available for every specifiable point. Forecasts and weather data are made in the more human-centric terms of towns and regions.

Severity: The problem will be faced by all classes of users in any task with this interface. It will prevent them from accomplishing many of the tasks they will want to perform, and will lead to users abandoning the program altogether.

Recommended Action: Allow the users to chose based on their own terms, or provide another mechanism for changing the location displayed. If the map only being moved in small increments, then a direct manipulation mechanism such as panning or scroll bars would work well. If large changes are expect, then other mechanisms such as hierarchical menus or name completion should be examined. Alternate mechanisms that are as efficient and compact as latitude and longitude are possible will still being usable to most people.

Problem: The system does not support habits and patterns of behaviour. Users of the system will have a number of locations that they are interested in, but the system provides no meaning of quickly moving between these locations. Further, there is no support for any sort of repeated monitoring behaviour such as tracking “what the weather will be like in Banff tomorrow afternoon” for every day of the week.

Severity: This will affect the intermediate and expert user as they attempt to use the system as more than a novelty to actually helping them plan and understand.

Recommend Action: Provide a history/bookmark/favorite places feature.

D.7 No Help

Problem: The current interface is inscrutable, and there is no means for the user to get a explanation of what the various control expect, how to interpret the error messages, or what the content of the display means. Documentation does not fix bad design, and all the problems previously documented will still exist.

Severity: We believe that this is not a major problem as TRAVELweather should not require much in the way of documentation.

Recommended Action: We recommend a redesign that eliminates the need for any documentation beyond a tutorial or brief demonstration. If the other problems identified in this report are correct, we believe the end result will be such an interface. Further, the amount of effort needed to fix most of the problems is no greater than the amount of effort needed to document the existing interface and will lead to greater profits and cost savings than producing the documentation.

Appendix E: PReSS Study Summaries

E.1 Study Summary for Participant #1

E.1.1 Participant and Task

This study involved a single participant who was familiar with the ideas of results synthesis. The participant had also taken part in a earlier results synthesis study using a paper based approach which was conducted some months earlier. The data set for the study is those problems reported for that earlier session. The participant had no contact or review of the problems, nor familiarity with the interface under review. The participant was given a description of the interface for reference. The participant received instruction in the capabilities of the interface. The study lasted about one hour. The participant did not complete the task due to time constraint, but made significant progress and was happy with the system and his work. He also felt that he could have completed the task with more time.

E.1.2 The System

E.1.2.1 What worked:

On the whole the system performed well. It was stable, and many of its features were exploited.

- The piles were used for grouping duplicates and reducing clutter/complexity. (ref. participant remark)
- Emergence occurred - higher level considerations that unified or otherwise went beyond what was in the detailed data. New connections were made, the “right level of abstraction” was sought, and achieved.
- Primarily used the overview for navigation.
- List view used mostly for organizing, as a drop target.
- Spatial layout was used, though not to the degree or subtlety seen in paper

E.1.2.2 What didn't work:

- Fixed sized space lead to crowding and awkward use of space.
- Problems with editing text annotations because you could only add or remove at the end - there was no ability to make edits in the middle of the annotation
- Desire to record info that didn't quite fit into any of the pre-existing categories (TA, PR) of input. This is a rich one in terms of the many possible ways it might be supported, and why the existing categories are not adequate.
- Many unintentional freehand annotations
- Expressed desire to more closely associate PD's with PR's
- Utility of overview reduced by inability to observe underlying content.
- Underlying objects of aliases not referenced i.e. never looked in full content window (though in this data set there is no clarification)
- Pile merge was desired but absent.

E.1.2.3 Unresolved Questions:

- Whether there was too much of a rush to certainty/simplicity/ organization.
- How to provide useful content overview/search
- How to communicate/record varying levels of done-ness and grouping (related but different).
- How to deal with complexity without removing subtlety and uncertainty - i.e. does allowing users to hide things away cut down on emergence?
- What is to be preserved in publishing? Just the problem reports or...?
- What is the right/best font size trade-off between readability and quantity of elements displayed

E.1.2.4 Miscellaneous Observations:

- The participant started out by making local organizations within initial heuristic groupings. These proved fruitful for identifying duplicates, which were put into piles. The participant focused on working

with groups that were small, avoiding the “big” until the last possible moment. After organizing a few heuristic groups, the participant began to create his own groupings, which really didn’t change. He also attempted to separate the things he had organized from those he had not. There was some sort of threshold effect as to when a group was created, once enough PD’s had accumulated, or the participant was convinced enough, then a problem report would be created which accounted for the grouping. When going through subsequent unexamined initial groups, more elements were adding to the already established groupings, and the participant indicated that he did not believe that the newcomers would change his interpretation or structuring.

- There was no monolithic familiarization step, but rather it was mixed in, done in the small on a group-by-group basis. Though I'm not sure if this was a conscious choice (partly, according to comments).

E.1.2.5 Changes resulting:

- Make group select default action, require key combo for FA.
- Add info area for fly-over in overview, maybe main view too.
- Add pile-merge capability

E.1.2.6 Prioritized System Errors:

- Deleting TA causes errors in list view
- Drop to create pile not working
- Could not edit text annotation
- word wrap in ProbRep
- scrollbar in ProbRep
- could not drag aliases on startup
- cursor in wrong visual state indicator
- editing not up to snuff in PR (selection deleting)
- empty text annotation created
- PR menu absent

E.1.3 The Study

E.1.3.1 What worked:

- Got some good observations and things to discuss.
- Found bugs and things to change.
- Participant knew what to do and didn't have a problem with the process.
- Got some good comments. Met the goals of the study

E.1.3.2 What didn't work:

- Forgot to explain group select and piles initially.
- To much fiddling to get things started, in particular the keyboard driver and the interface description.

E.1.3.3 Unresolved questions:

- Do I swamp them with stuff to do and too many ways of doing things?
- How much should I explain why/how to use features on a task based level, and how much should I let them figure it out for themselves?

E.1.3.4 Changes resulting:

- Have participants read the interface description first.
- Instruct the participant about piles and group selecting. And that group selecting works in the overview.

E.2 Study Summary for Participant #2

E.2.1 Participant and Task

This study involved a single participant who had previous exposure to results synthesis. The participant had also taken part in a earlier results synthesis study using a paper based approach which was conducted some months earlier. The data set for the study was the same as that used for study #1. The participant had performed heuristic evaluations of the interface in the past, but not recently. The participant was given a description of the interface for reference. The participant received instruction in the capabilities of the interface. The study lasted about one hour. The participant did not complete the task due to time constraint, but made progress and was happy with the system and his work. He also felt that he could have completed the task with more time.

E.2.2 The System

E.2.2.1 What worked:

On the whole the system performed well. It was stable, and many of its features were exploited.

- The piles were used for grouping cards in a “same problem” relationship
- The participant moved cards out of the initial groupings.
- The participant found “underlying” patterns or commonalities in the data.
- Spatial layout was used, though not to the degree or subtly seen in paper. In particular, the x axis was significant, but the y axis was not meaningful in the layouts created by the participant.
- The initial layout was judged to be good in so much as it allow easy reduction in complexity
- The overview-info window was used extensively to search the workspace. The participant remarked on its utility in the debrief.

E.2.2.2 What didn't work:

- Panning in the main view was difficult due to perceived mismatch between input and reaction - the movement was “swirly.”
- The participant did not create any text annotations, though in debrief it became apparent that he could have used them, but did not think of it.
- There were a couple of unexplained error messages early on that did no recur and had no effect on system functionality.
- In debrief, the participant indicated a strong desire for an overview that would allow him to read all the cards at once.
- Cloning was not used.
- Participant tried to move free-hand annotations.

E.2.2.3 Unresolved Questions:

- What alternate view mechanisms might be better? (Zooming, fisheye, dragmag,...)
- What is the best initial layout for the problem descriptions?
- What is the best way to reduce the apparent complexity, or have people tolerate it better?

E.2.2.4 Miscellaneous Observations:

- The participant started out by making local organizations within initial heuristic groupings. The participant overwhelming concern was to reduce the complexity of the workspace so he could get a better overview.
- At the outset, the participant was intimidated by the apparent complexity of the dataspace and stated that in the beginning he was only going to do local organization and could not do any “cross-referencing” between the groupings. Later in the process, he did move cards between groupings.
- The participant spent a lot of time reviewing the data – looking for something.

- The participant did a lot of “micro” organization within the initial groupings. He would separate out items he thought related, and once he had confirmed their relationship in his mind, he would put them “back”, and often into a pile.
- The participant stated that cards not in piles but close by were there because he was uncertain as to where to put them – they were related but not quite close enough, or they might belong in one or more groups.

E.2.2.5 Changes resulting:

- Add ability to add user content to workspace (notes)
- Change button for panning in overview
- Make functionality more visible

E.2.2.6 Prioritized System Errors:

- Panning in main view “hard.”
- Make free-hand annotations moveable and selectable
- Group select not behaving as expected
- Spurious error message about deselecting

E.2.3 The Study

E.2.3.1 What worked:

- Got some good observations and things to discuss.
- Found bugs and things to change.
- Got some good comments. Met the goals of the study
- I had everything setup and ready to go at the beginning of the study.
- 8mm camera seemed to work well.

E.2.3.2 What didn't work:

- Did not fully explain all functionality until study underway, especially with respect to group selecting.

E.2.3.3 Unresolved questions:

- How did not authoring any of the PD’s effect the process?
- How much should I explain why/how to use features on a task based level, and how much should I let them figure it out for themselves?

E.2.3.4 Changes resulting:

- Prepare and use script for instructing participants in system functionality.

E.3 Study Summary for Participant #3

E.3.1 Participant and Task

This study involved a single participant who had previous exposure to results synthesis. The participant had also taken part in a earlier results synthesis study using a paper based approach which was conducted some months earlier. The data set for the study was the same as that used for study #1. The participant was one of those involved in the exercise that generated the data. The participant had performed heuristic evaluations of the interface in the past, but not recently. The participants professional background is as a software developer. The participant was given a description of the interface for reference. The participant received instruction in the capabilities of the interface. The study lasted about one hour. The participant had organized the data and had generated corresponding problem reports at the end of the hour, but indicated that he could do further work.

E.3.2 The System

E.3.2.1 What worked:

On the whole the system performed well. It was stable, and many of its features were exploited.

- The piles were used for grouping cards in a “same problem” relationship
- The participant moved cards out of the initial groupings.
- The participant found patterns or commonalities in the data that were not reflected in the heuristic groupings.
- Spatial layout was used, though not as much as I had hoped, nor the level at which I had expected.

E.3.2.2 What didn't work:

- Text annotations proved problematic - in some cases generating errors, and in other cases just frustrating the participant or generating unexpected results. The participant expected visual feedback for when it was “OK” to type. Also at times mouse movement caused unexpected editing results - I interpret this to mean that the participant expected much more strongly moded interaction. The participant also created a number of blank text annotations.
- The participant used the TA list very little, or not at all.
- The participant wanted to be able to “drop” the problem descriptions into the problem reports - form a closer and more formal bond.
- Cloning was not used. The participant stated that he thought it would be of more use in group situations.
- The participant did not use the shuffle button on the piles, opting instead for the menus.
- The participant, in the beginning, would look for functionality in the system menus rather that the context menu.
- The participant had a hard time selecting the problem reports in order to move them.

E.3.2.3 Unresolved Questions:

- How do I resolve clashes between theory and practice? (TA mechanisms, PD/PR relationship/visibility)
- How do I deal with people who want to use the tool in ways I think are non-optimal?
- Readability vs. amount of content on screen.

E.3.2.4 Miscellaneous Observations:

- The participant started creating problem reports very early in the process.
- The participant had definite opinions about how the data should be organized - what the right way of doing things was. And this was related to conventional notions of creating bug reports for developers.
- The participant started out reviewing the problem descriptions as in a “conventional” familiarization step, but about half way through announced that he was going to start organizing things because, among other things, he was “bored.”

E.3.2.5 Changes resulting:

- I am undecided about making any additional changes. I am considering implementing a more moded style of interaction.

E.3.2.6 Prioritized System Errors:

- Can’t create TA when problem report on screen, or maybe it is that you can’t create a TA immediately following creating or adding content to a PR.
- TA focus fragile - hard to edit, easy to unintentionally create new ones
- Spurious error message about deselecting
- Make moving PRs easier
- Two insertion cursors present on screen at times

E.3.3 The Study

E.3.3.1 What worked:

- Got some good observations and things to discuss.
- Found bugs.
- Met the goals of the study.

E.3.3.2 What didn't work:

- The participant noted the absence of informed consent.

E.3.3.3 Unresolved questions:

- To what degree do I want to enforce process?
- Should I run more individuals? What would I hope to learn?

E.3.3.4 Changes resulting:

- Create consent and instruction sheets.

E.4 Study Summary for Groupware #1

E.4.1 Participant and Task

This study involved a two participants. One was the experimenter, who is an expert on results synthesis. The other participant had received training on heuristic evaluation in the past, but no exposure to results synthesis. The data set for the study was the same as that used for study #1. The participant had no familiarity with the data set or the evaluated interface. The participants professional background is as a graduate student in software engineering. The second participant was given a description of the interface for reference, and allowed to read it before commencing the task.. The second participant received instruction in the capabilities of the interface. The study lasted about one hour. The participants had performed significant reorganization of the data, but did not complete the task, though they felt that they would given more time. Both participants were experienced users of real-time groupware.

E.4.2 The System

E.4.2.1 What worked:

On the whole the system performed well in its first use as groupware. It was stable, and many of its features were exploited.

- The piles were used for grouping cards in a “same problem” relationship
- The participants moved cards out of the initial groupings, establishing new consensual groupings.
- The participants were able to work both apart and together, with relative ease in transition.
- The participants were able to make diectic references to the workspace that the other participant could interpret
- The participants found patterns or commonalities in the data that were not reflected in the heuristic groupings.
- Spatial layout was used.
- Cloning was used a fair bit, though they tended to end up in the same spot after further work.

E.4.2.2 What didn't work:

- P2 requested a way to get rid of “unnecessary” clones.
- P2 requested a textual search function.

E.4.2.3 Unresolved Questions:

None.

E.4.2.4 Miscellaneous Observations:

None.

E.4.2.5 Changes resulting:

None.

E.4.2.6 Prioritized System Errors:

- Telepointers were often “under” and object.
- Text annotations could not be group-selected in the main view.
- Object view windows caused “problems” - spurious error messages about selection and not being able to move cards “temporarily.”
- The edit option on the text annotation pop-up menu is superfluous.
- Spurious errors in trying to select and move a heterogeneous group of elements in the system demo.

E.4.3 The Study

E.4.3.1 What worked:

- Got some suggestions about how to improve the interface.
- Found bugs.
- Met the goals of the study.

E.4.3.2 What didn't work:

- P2 started out suggesting things be re-arranged under the heuristic categories. It was a misunderstanding of the process.

E.4.3.3 Unresolved questions:

None.

E.4.3.4 Changes resulting:

None.

E.4.3.5 Misc. Comments:

- This was supposed to be mainly a bug hunt, and on that count it was successful.
- It was hard to both participate and observe at the same time - note taking suffered.
- I tried to minimize my influence as someone who was familiar with the process, data, and system. But I have no idea how well that worked.

E.5 Study Summary for Groupware #2

E.5.1 Participant and Task

This study involved a two participants. One was the experimenter, who is an expert on results synthesis. The other participant had performed heuristic evaluation, but had no training in it nor in results synthesis. The data set for the study was a set of problem descriptions generated specifically for the groupware studies. This data set was based on the same interface as that used in the preceding studies, but shared no contributors or problem descriptions in common. P2 was one of the contributors to the data set. P2’s background is as a graduate student in computer science. P2 received instruction in the capabilities of the interface. The study lasted about one hour. The participants had performed significant reorganization of the data, but did not complete the task, though they felt that they would given more time. P2 had not used real-time groupware before.

E.5.2 The System

E.5.2.1 What worked:

On the whole the system performed well. It was stable, and many of its features were exploited.

- The participants moved cards out of the initial groupings, establishing new consensual groupings that were more clear than the original ones.
- The participants were able to work both apart and together, with relative ease in transition.
- The participants were able to make diectic references to the workspace that the other participant could interpret
- The participants found patterns or commonalities in the data that were not reflected in the heuristic groupings.
- Spatial layout was used.

E.5.2.2 What didn't work:

- P2 started out moving aliases into her view instead of moving the view. The middle mouse button is not often used, and there aren't many indications of the functionality in the interface - it has to be remembered.
- P2 complained about small fonts in the main view.

E.5.2.3 Unresolved Questions:

- Should RS on the system be different than on paper? Am I being too slavish to the paper process?
- I have this idea that those more comfortable with the task/interface are better able to pay attention to what is going on (Ref. action resources).

E.5.2.4 Miscellaneous Observations:

- There was a definite learning curve in P2's interaction with the system. As the study progressed, the participant used more of the functionality with more fluency.
- To avoid view moves the participants would put aliases they wanted to discuss into the view of the other person.
- People should probably not learn the interface and RS at the same time. Perhaps paper RS should be a prerequisite to using the system.

E.5.2.5 Changes resulting:

None.

E.5.2.6 Prioritized System Errors:

- One unreproduced error having to do with selecting and moving at the very beginning.

E.5.3 The Study

E.5.3.1 What worked:

- Got some suggestions about how to improve the interface.
- Met the goals of the study.
- There was "2nd-round" organizing - reconsideration and uncertainty

E.5.3.2 What didn't work:

- Learning the interface got in the way of performing the task.
- P2 didn't understand/feel comfortable with the task until some ways into it.

E.5.3.3 Unresolved questions:

- Does the study last long enough for emergence to really occur?

E.5.3.4 Changes resulting:

- Provide participants time to play with the interface before starting the main task.

E.5.3.5 Misc. Comments:

- Things proceeded basically as expected.
- It was hard to both participate and observe at the same time - note taking suffered.

- I tried to minimize my influence as someone who was familiar with the process, data, and system. But I have no idea how well that worked. There is also a question of cultural biases and linguistic problems.

E.6 Study Summary for Groupware #3

E.6.1 Participant and Task

This study involved a two participants. Both participant had been recently introduced to heuristic evaluation and had performed heuristic evaluation in preparation for the study. Neither had training in heuristic evaluation or in results synthesis. The data set for the study was a set of problem descriptions generated specifically for the groupware studies. This data set was based on the same interface as that used in the preceding studies, but shared no contributors or problem descriptions in common. Both participants were contributors to the data set. P1's background is as a professional graphic/interface designer and Mac user. P2's background is as a human factors practitioner. Both participants received instruction in the capabilities of the interface, and a few minutes to practice with the interface. The study lasted about one hour. The participants had performed significant reorganization of the data, but did not complete the task, though they felt that they would given more time.

E.6.2 The System

E.6.2.1 What worked:

On the whole the system performed well. It was stable, and many of its features were exploited.

- The participants moved cards out of the initial groupings.
- The participants were able to work both apart and together.
- The participants were able to make diectic references to the workspace that the other participant could interpret
- The participants found patterns or commonalities in the data that were not reflected in the heuristic groupings.
- Spatial layout was used, but at a very simplistic level.
- Both participants were basically comfortable with the interface and able to work productively with it.
- Piles were used to reduce clutter, and were considered a natural mechanism.
- The participants liked the overview, especially after I remembered to tell them about "hover help."

E.6.2.2 What didn't work:

- P1 was confused by the view-pan in the main view as he was used to document-pan.
- Both participants commented on/complained about small fonts.
- The participants would occasionally try to move in the overview by dragging with B1 instead of B2, but they remembered without prompting.
- The participants used problem reports as their landmarks/category headings. This lead to missing/desired functionality that was present, if they had used TAs instead. As a result, they thought TAs were largely redundant.
- I forgot the binding for freehand annotations, which P1 wanted to do.
- P1 wanted to be able to see all the parts of a problem report at once.
- The participants commented on not thinking to look at the complete text object.
- The participants did not used the TA list.
- Text selection weirdness (minor point).
- P2 found the use of main view pan to be disorienting (lag in update).
- P1 was anxious about the possibility of the work being lost in a crash (no autosave).

E.6.2.3 Unresolved Questions:

- Process education/enforcement.
- In-place "hover help?"

E.6.2.4 Miscellaneous Observations:

- There was a (voluntary/implicit) sharp division of activities towards the beginning of the task. P1 would create problem reports with skeletal text and move aliases into problem groups, while after initial moving around, P2 spent most of here time elaborating the problem reports. P2 latter commented that she was much quieter than “usual” (and uninvolved?). P2 latter became more involved in the process.
- The organization process was to noticed something repeated, create a group “heading” for it, and then search for things that should be members in it.
- The participants mentally partitioned the space into “done” and “not done” areas, and sought separation between the two. (Ref. use of probReps).
- The participants started out working independently – there was a distinct pause some way into the task as they “check out” what the other had done.
- I’m not sure exactly when and where I observed this, but a number of participants had an aversion to having overlapped aliases.
- The aliases associated with completed problem reports were arranged in a very orderly rectilinear fashion.
- The participants would occasionally put an alias in the view of another for consideration.
- P2 suggested that working face to face was faster due to more facility with manipulating and monitoring physical items (two handed manipulation).
- P1 requested the ability to partition space other than white space.

E.6.2.5 Changes resulting:

- Change problem report to be outline. (Not implemented).
- Change hover to deal with long content. (Not implemented)
- Enable hover for probReps. (Not implemented).

E.6.2.6 Prioritized System Errors:

- FA key binding obscure.
- The problems were not initially distributed as evenly about the workspace as I expected.
- P1, on joining the practice session, did not have the text annotations.
- There was a non-reproduced error about ws not found in the practice session.

E.6.3 The Study

E.6.3.1 What worked:

- Met the goals of the study.

E.6.3.2 What didn't work:

- P1 would have preferred to be able to see the other participants face(s).

E.6.3.3 Unresolved questions:

None.

E.6.3.4 Changes resulting:

- Explain the scenario for the study (not implemented)
- Explicitly suggest the use of TAs as labels/landmarks.

E.6.3.5 Misc. Comments:

- One of them was tired, the other quiet.

E.7 Study Summary for Groupware #4

E.7.1 Participant and Task

This study involved a two participants. Both participant had been recently introduced to heuristic evaluation and had performed heuristic evaluation in preparation for the study. Neither had training in heuristic evaluation or in results synthesis. The data set for the study was a set of problem descriptions generated specifically for the groupware studies. This data set was based on the same interface as that used in the preceding studies, but shared no contributors or problem descriptions in common. Both participants were contributors to the data set. P2's background is as graduate student in Environmental Design and a Mac user. P1's background is as a honours psychology undergraduate. Both participants received instruction in the capabilities of the interface, and a few minutes to practice with the interface. The study lasted about one hour. The participants had performed significant reorganization of the data, but did not complete the task, though they felt that they would given more time.

E.7.2 The System

E.7.2.1 What worked:

On the whole the system performed well. It was stable, and many of its features were exploited.

- The participants moved cards out of the initial groupings though not very far.
- The participants were able to work both apart and together.
- The participants were able to make diectic references to the workspace that the other participant could interpret
- The participants found patterns or commonalities in the data that were not reflected in the heuristic groupings.
- Spatial layout was used.
- Cloning was used extensively.
- Both participants were basically comfortable with the interface and able to work productively with it.

E.7.2.2 What didn't work:

- Commented on small fonts - leaned in close to the screen.

E.7.2.3 Unresolved Questions:

- Action tracking – feedback on/auditing of actions of other participants. Relates to issues of trust, group familiarity.
- Eliminate heuristic labels?
- Hiding/associating aliases with problem reports

E.7.2.4 Miscellaneous Observations:

- The participants never made use of the empty lower part of workspace but organized "in place."
- P2 requested the ability to edit aliases because spelling and grammar errors "disturbed" him.
- Participants suggested the actual interface as the background for the task.
- Participants removed clutter by piling.

E.7.2.5 Changes resulting:

- Better communication of selection/focus of attention of other participant (Not implemented). In particular, tracking of drop to TA list was mentioned. Might this be a side effect of their congested space?
- Clone top of pile without removing it (Not implemented).
- Provide mechanisms for view slaving/synchronizing (Carl's failed mechanisms).
- A way to track clones was requested. (Not implemented)

E.7.2.6 Prioritized System Errors:

- Displayed pile top was not the same for both instances.

- Ability to clone groups and piles.
- Allow piles in group select “Put in Pile.”

E.7.3 The Study

E.7.3.1 What worked:

- Met the goals of the study.

E.7.3.2 What didn't work:

- Perhaps excessive development focus, leading to a widget focus in organizing.

E.7.3.3 Unresolved questions:

None.

E.7.3.4 Changes resulting:

- Prepare instructions in interface and link with hands-on practice.

E.7.3.5 Misc. Comments:

None.

E.8 Study Summary for Groupware #5

E.8.1 Participant and Task

This study involved three participants. P1 had no previous experience with results synthesis. P2 was familiar with the study and had previously participated in both paper based results synthesis and one of the single user trials. P3 was the experimenter. The data set for the study was a set of problem descriptions generated specifically for the groupware studies by students in a graduate course. This data set was based on the same interface as that used in the preceding studies. None of the participants were contributors to the data set, but P3 had reviewed the data extensively. P1’s background is a professor and groupware researcher with a degree in engineering. P2’s background is as a professor and supervisor of the research. The study lasted about one hour. The participants had performed significant reorganization of the data, but did not complete the task, though they felt that they would given more time.

E.8.2 The System

E.8.2.1 What worked:

On the whole the system performed well. It was stable, and many of its features were exploited.

- The participants moved cards out of the initial groupings into new ones that resulted from review of and working with the data.
- The participants were able to work both apart and together.
- The participants were able to make diectic references to the workspace that the other participant could interpret
- The participants found patterns or commonalities in the data that were not reflected in the heuristic groupings.
- Spatial layout was used.
- Both participants were basically comfortable with the interface and able to work productively with it.

E.8.2.2 What didn't work:

- Resynchronizing after individual work required noticeable effort and delay, though it did not bring about any noticeable task tailoring.
- Participants mentioned not being able to figure out specifically what other participants had done during times of loosely coupled work.

- The tabbed dialogs were problematic. Problems arose when more than one person wanted to look at the content.
- Participants had trouble mapping between on-screen representations and participants (Where are you?/ Which one are you?)
- Participants had trouble seeing when other participants were operating on things they were dealing with (P2: “oops” when P3 unpiles recently piled aliases).

E.8.2.3 Unresolved Questions:

- How do I present/preserve the actions of the participants (traces or trails or replay) so that the other participants could figure out what had happened when they weren’t closely tracking the other person’s activity.
- How do I deal with large text spaces? Locking? Relaxed views?

E.8.2.4 Miscellaneous Observations:

- P3 found the use of telepointers distracting early in the familiar stage – there was a lot of irrelevant movement in the visual field.

E.8.2.5 Changes resulting:

- P2 requested drag scrolling (Not implemented).
- Space “warping” – the ability to give a group more space in place without having to move it to a new location – things were getting crowded on occasion (Not implemented).

E.8.2.6 Prioritized System Errors:

- Spurious error about unknown deselect element when trying to move group.

E.8.3 The Study

E.8.3.1 What worked:

- Met the goals of the study.
- Participant unfamiliar with the system was able to navigate and participate.

E.8.3.2 What didn't work:

None.

E.8.3.3 Unresolved questions:

- More 3 user or greater multi-user trials?

E.8.3.4 Changes resulting:

- Provide participants with a better sense of context within which they are performing the results synthesis – where the interface is in the development cycle, some sense of the receptiveness of developers and designers and the resources available to fix the problems.

E.8.3.5 Misc. Comments:

None.