Knowledge–Based Assistance for Accessing Large, Poorly Structured Information Spaces

by

Curt Stevens

B.A., University of California, Berkeley, 1984

M.S., University of Colorado, Boulder, 1989

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirement for the degree of

Doctor of Philosophy

Department of Computer Science

1993

This thesis for the Doctor of Philosophy degree by

Curtis Frank Stevens

has been approved for the

Department of

Computer Science

by

_____

Gerhard Fischer

_____

Clayton Lewis

Date _____

Abstract

Stevens, Curtis Frank (Ph.D., Computer Science)

Knowledge–Based Assistance for Accessing Large, Poorly Structured Information Spaces

Thesis directed by Professor Gerhard Fischer

Large information spaces present several problems to people searching for interesting information including information overload. This thesis describes a novel approach to handling overload problems in the domain of Usenet News, an open access computer–based bulletin board system that distributes messages and software. A conceptual framework is developed that shows the need for (a) flexible organization of information access interfaces, (b) personalized structure to deal with vocabulary mismatches and individual information needs, and (c) semi–autonomous agents that assist in creating this personalized structure. In addition, an operational system (INFOSCOPE) instantiate this framework allowing for the exploration and evaluation of the approach in realistic working environments. Using INFOSCOPE, Usenet readers evolve the predefined system structure to suit their own tasks or semantic interpretations. Agents assist users by suggesting filters based on observed interest patterns. Evaluation of the system indicates that (1) personalized structure is quickly adopted and effectively used; (2) personalized structure reduces information overload caused by uninteresting messages; and (3) INFOSCOPE agents are an effective aid to the creation of personalized structure in Usenet News. However, user studies have exposed deficiencies in the current implementation that indicate a more sophisticated suite of agents could increase the usefulness of many suggestions.

Also available from the University of Colorado at Boulder Computer Science Department as Technical Report # CU–CS–640–93.

for Carol

your love and support are
the greatest discoveries of all

I love you

Acknowledgements

I would like to thank the following people:

Gerhard Fischer for patiently allowing me to find my direction in a sea of potential research issues. Also, for allowing me easy access to the guidance, criticism and experience needed to generate worthy research. Finally, for providing an environment for research that was both stimulating and fun.

Each of the members of the Human–Computer Communication research group (past and present) for helping mold my research through discussions. Thanks especially for tearing me up at numerous HCC meetings.

Andreas Girgensohn for helping with areas of implementation meant only for the most serious hackers in the universe. Without you there would be no INFOSCOPE.

My thesis committee, past and present, Gerhard Fischer, Walter Kintsch, Clayton Lewis, Jim Martin, Mike Eisenberg, Ray McCall and Michael Schwartz for helpful commentary and keeping the jokes about how long this took to a minimum.

All of the anonymous testing subjects for their suggestions and graceful donation of time and effort to this research.

My parents, Carole and Norman, for their incredible support over the years. Without you there would be no research, no thesis, and no Doctorate. I love you both and I owe you everything.

Norman Stevens, my dad, for lending me the money to buy my Mac.

The Army Research Institute (ARI grant MDA903–86–CO143) and Gerhard Fischer for funding this research.

Contents

Chapter

Figures

Tables

1. Introduction and Overview

As global networks expand their coverage to more sites and make information on an ever widening array of topics available, they are serving as useful high bandwidth communication channels between users. The amount of electronic information available to people on their desktops is more than they could ever hope to process. One global network that amply represents this problem is the Internet [Quarterman, Hoskins 1986], specifically in its distribution of Usenet News [Horvitz 1989; Kantor, Lapsley 1986]. Because of how the News distribution system accepts new messages, and the huge number of messages, users find themselves constantly confronted by the following problems associated with large poorly structured information spaces. When users wish to distribute news (messages) through Usenet, they must first choose a *newsgroup* into which the message is classified. Readers then find the message in that newsgroup. However, the semantic classifications (vocabulary etc.) of the sender are not necessarily the same as those of the thousands of readers who might read this message. A small study I conducted verified this classification problem. Users classified the identical messages into several different newsgroups. Since the process of selecting a newsgroup classification can be difficult for the user, a somewhat general classification hierarchy has been established in Usenet News (see Figure 1.1). It follows that each newsgroup potentially contains messages spanning a wide range of topics. Particular readers of a newsgroup may be forced to browse many messages they deem irrelevant in order to find information they deem interesting. Another empirical study that I carried out demonstrates that users only subscribe to a fraction of the newsgroups they find interesting for this very reason. The size of the information space often frustrates the users' ability to effectively utilize the available information. Since messages expire at regular intervals the problem is even worse. The transitory nature of the data makes it crucial that readers can find relevant information in a timely manner.

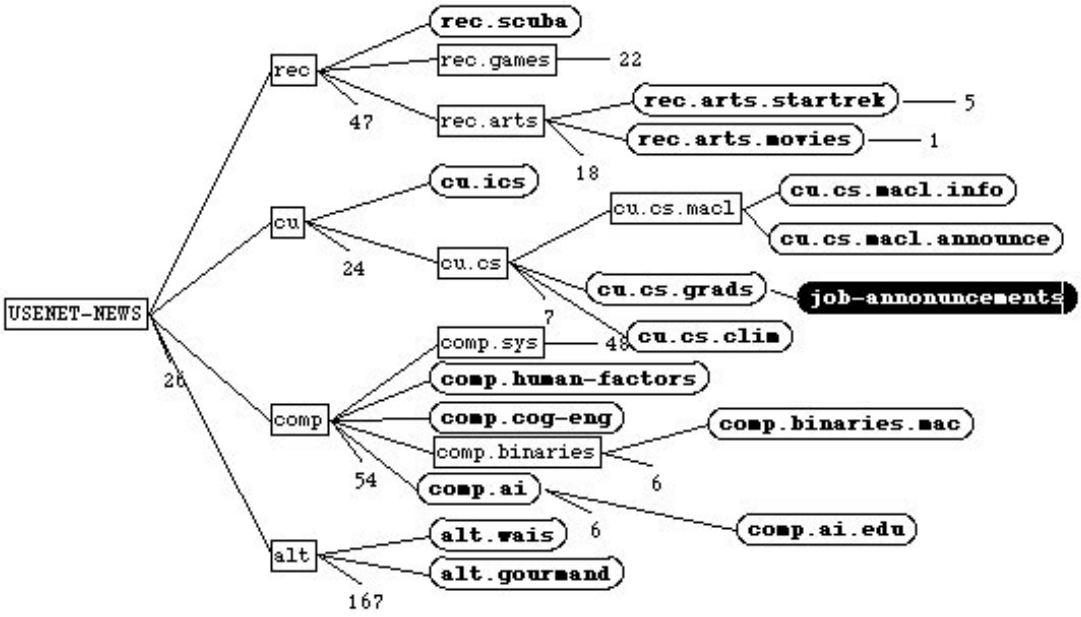


Figure 1.1 The Usenet Hierarchy

Here is a partial cross–section of the total Usenet hierarchy. Even though there are nearly 2000 Usenet newsgroups (the clear oval nodes in this picture), the topics are general enough that people can decide where to post messages. Because many newsgroup names can overlap, messages are often posted to more than one newsgroup. The dark oval nodes represent user–defined extensions to the standard Usenet hierarchy.

This thesis describes research that exposes and compensates for inadequacies in the current generation of electronic messaging systems. What follows is a discussion of the INFOSCOPE project which explores methods of helping alleviate problems associated with the information overload encountered when accessing Usenet News [Fischer, Stevens 1991]. In Chapter 2 the problem is introduced. The domain is discussed in Chapter 3. Following that is a deeper discussion of the motivations for this research in Chapter 4. A conceptual framework and background for the project is then developed in Chapter 5, followed by a discussion of related work that influenced this project in Chapter 6. In addition, an approach to the described problems is discussed by introducing the INFOSCOPE prototype system through a scenario in Chapter 7 and a discussion of the design methodology in Chapter 8. The evaluations and their results are covered in Chapter 9. Finally, the limitations of this approach, prospective future work and a summary are presented in Chapters 10 and 11.

2. Problem Statement

The existence of message distribution systems such as Usenet News allow users from distant parts of the world to freely communicate, but it creates difficult problems for users of the systems that access these messages. When accessing news, users must constantly perform mappings from their own personal semantics, in which their interest in information is based, to the semantics of a predefined hierarchy of newsgroups. This is further complicated by the fact that the messages are classified by the sender of the message, requiring a guess on the part of the reader as to where someone might classify information on a specific topic of interest. For example, a reader looking for information about the EMACS text editor on the Macintosh might browse any or all of the newsgroups comp.emacs, gnu.emacs, comp.text.desktop, comp.editors, or the related Macintosh based newsgroups comp.sys.mac.misc, comp.sys.mac.digest, or comp.sys.mac.apps (for applications). The difficulty of this process depends upon the degree of similarity between the semantic interpretation chosen by the message sender and that chosen by the reader. The wider the gap between these interpretations, the more difficult a reader may find this task. Remember, however, that once a message is sent the sender doesn't have to find it but the reader does. This leads to the need for reorganizing the information space based upon the personal semantics of message readers, not message senders. This research is based on the hypothesis that semantic organization of newsgroups should be centered in the semantics of message readers not message senders. Specifically, readers should be able to modify and extend the predefined newsgroup hierarchy with respect to classifications. Extending the previous example, for the duration of users' interest in EMACS they should be able to define a single newsgroup that contains all information about that program. Alternately, a particular user might wish to define a newsgroup which contains only those messages mentioning both EMACS and the Macintosh. Unfortunately, personalization of this type is a difficult process that has been examined in work on adaptable systems and end–user modifiability [Fischer, Girgensohn 1990]. This difficulty leads to the conclusion that users need help in carrying out personalization. This is especially true for users who are interested in many different topics over long periods of time. Keeping track of these interests as they evolve and disappear may require frequent redefinition of these personalized structures. Providing assistance for this process will allow users to concentrate on finding interesting messages.

3. The Domain: Usenet News

Usenet News is not a computer network, but a distribution medium that takes advantage of the existence of networks. This section explores the differences between these two concepts and briefly discusses the characteristics of Usenet News.

3.1 Networks

Computer networks are groups of computers using common protocols for communication among themselves [Quarterman, Hoskins 1986]. The use of standard protocols allows the machines on a network to be heterogeneous, so long as they all utilize the same protocols. Common protocols include ethernet, X.25, TCP/IP, and UUCP among others. Computers communicate through many of these protocols simultaneously in order to get the benefits of the services (text transfer, binary transfer, asynchronous interprocess communications, file sharing, address verification etc.) provided by each. In addition to containing various types of computers, networks are connected by various communication mediums including direct connection, connection via satellite, and connection via phone line (modem). The availability of modem connections allows people to connect to networks from the personal computers on their desktops. For example, personal computers with software that uses the UUCP protocol can be members of networks based on that protocol, provided that a current member of the network is willing to make access available. Similarly, computers on ARPANET each communicate via TCP/IP.

When several networks are each based on the same protocol, an *internet* can be formed by connecting communication channels based on that protocol. An example of this is the ARPA internet that connects several large networks including ARPANET. The ARPANET in turn participates in CSNET, itself a collection of several networks. The proliferation of peripheral connections between various networks creates a world wide meta–network.

Machines on a network are identified by *addresses*. These addresses can be numbers, but are commonly names chosen by the owner of the machine. By specifying that text should be sent to a specific user at a specific machine (e.g., stevens@cs.colorado.edu), a combination of address verification and text transfer services essentially sends a message. By writing software which exploits these capabilities, complex communication channels allow users to exchange messages across the network.

3.2 Messages and Distribution Mediums

Distribution mediums are collections of software that, when standardized and widely accepted, allow users anywhere on a network to communicate via *messages* and transfer generic data. Messages contain headers and body text. Header fields are located at the top of a message and contain information about the destination of the message (e.g., its address), the subject of the message, who it is from (a return

address), and whatever other information is needed by the distribution medium in use. The body text contains the actual message, which can be anything from a simple memo to encoded binary applications. Almost all of this is simply convention, however, based on common protocols. For example, the ARPA internet definition of a message is anything with a subject header field. No additional header fields or body text is required for it to be considered a message. However, it is the extra information required in message headers by these protocols which constitutes the small amount of structure necessary to provide services like electronic mail.

Using several header fields, messages contain enough structure to allow for organized distribution. The degree of this organization depends upon the distribution medium being used. The simplest of these is the most common, electronic mail. Proper distribution of email requires that messages contain a subject (required by the internet), a destination, and a return address (so that replies to a message will be properly addressed). In addition to this, many systems add other fields like summaries of the body text (completed by the sender) and the number of lines in the message (completed by the software system).

Soon after email systems proliferated, people noticed that they often sent the same message to a predetermined group of addresses. In response to this need, email systems added the ability to define *mailing lists*. After defining an *alias* representing a group of recipients, a sender need only specify the alias as the address of a message. The distribution medium expands the alias after the message is sent, ensuring that each member of the mailing list receives their own copy of the outgoing message. For example, in our research group there is an alias ‘hcc’ which sends a message to all members of the Human Computer Communication group. Senders simply address their messages to hcc. As the members of the group change, senders need not modify their actions since the alias will be changed to reflect new members of the mailing list.

Another type of distribution medium is the bulletin board system (bbs). Instead of distributing information by explicitly sending it to a list of other users, bbs’s serve as a central repository of information. Users utilize network connections to access the bbs and browse the available information. Those messages or data that interest users can be downloaded to the user’s local computer system. In this way messages need not be sent to anyone, and users that are not known to the bbs system or the message sender can connect and enjoy the benefits of the information repository. Due to the nature of this medium, it is commonly used to distribute public domain software, and to serve as archive sites for the messages sent out on important mailing lists. Examples of bulletin board systems include Compuserve and GEnie. UNIX systems commonly set up bbs’s by creating a user account that anyone can log into without a password. A limited environment based on ftp (file transfer protocol) allows users to browse and download information. Finally, many bbs’s are actually running on personal computers in people’s homes.

3.3 Usenet News

The Usenet News distribution medium is a combination of the medium types described above. Electronic mail and mailing list systems allow users to send messages

without actually connecting to other machines via a network. This is done by using certain protocols and letting the network automatically transfer the information based on addresses. Bulletin board systems, however, require a connection between two machines. Users actually log onto the remote system and interact with the system using whatever mechanisms are provided by the bbs. Each of these methods has its disadvantages. Mailing lists waste network bandwidth when they send the same message to several people on the same machine. They limit the recipients of general interest information to those contained in the alias. Bulletin boards address these problems at the expense of requiring an actual connection between two machines. If the connection cannot be made, users must wait before sending or receiving information. Mail, on the other hand, can be spooled for later delivery or rerouted around machines that are not operational.

For these reasons Usenet News combines characteristics of both mailing lists and bulletin board systems. A bulletin board system provides access for the purpose of reading, answering and sending messages. The bbs is distributed to many sites and users configure their computers to connect to the most convenient location. This represents a compromise between availability of connections and redundancy of data transmitted across the network. In order to further reduce the necessity to make connections to specific machines, the Usenet News medium allows message spooling just as in mail systems. Outgoing messages are proliferated to distribution sites as the connections become available. Since News is a broadcast medium, there are no addresses for messages. Instead, a header field is added to each message which specifies a destination newsgroup. Each newsgroup ideally delineates a set of topics. For example, rec.skiing includes messages relating to recreational skiing. Almost two thousand newsgroup classifications currently exist and they are organized into a hierarchy. Software displays messages from only one newsgroup at a time, effectively structuring the information based on these classifications. In this manner, thousands of messages a day are distributed and accessed across the network.

3.4 The Classification Hierarchy behind News

The Usenet News information space is organized around a hierarchy of newsgroups. Each node in the hierarchy represents either a specialization of the hierarchy or a newsgroup that contains messages. An example of several newsgroup names can be seen in Figure 1.1. A path through the hierarchy identifies a particular node and a ‘dot’ separates each element of the path. General categories in the hierarchy range in topics from *recreational* (rec.), to *computer* (comp.), to local categories like *Colorado* (co.) among many others. Since classifications are very general the software allows users to *cross–post* messages to several newsgroups at once. For example, a message about problems formatting this paper could potentially be sent to the newsgroups comp.sys.mac.apps, comp. desktop, and comp.windows.

3.5 A Usenet Culture

As mentioned earlier, the News domain differs from that of electronic mail in that messages are sent to central repositories rather than specific groups of individuals. This means that message senders no longer have control over who reads their messages. “Inside jokes” or colloquial expressions may be funny to one set of readers

but offensive to others. Problems like this have lead to the establishment of a Usenet culture. This culture recognizes, among other things, that there must be a certain etiquette maintained. For example, facetious remarks are often followed by a "smiley face :->" to indicate that no offense should be taken. Similarly, potentially offensive jokes are expected to be encrypted and news reading systems have the ability to decrypt these messages built right into the software. Failure to adhere to this etiquette produces consequences that range from negative feedback to public embarrassment to removal of network privileges.

An important question to ask in the Usenet domain concerns the incentive people have to participate in this free exchange of information. For example, why do users willingly expend the time and effort necessary to answer messages from novices with questions? One answer for this is that users of Usenet begin to feel as though they are part of a community. As long term, and thereby knowledgeable members of the community some users will expend extra effort for the sake of the community itself. Others participate after someone on *the net* posts a useful answer to a desperately asked question. Whatever the reason, the huge volume of messages serves as evidence that people do participate actively.

Established members of the Usenet community have recognized over time that new users to the community share difficulties and common questions about their new environment. For this reason there is a newsgroup named news.announce.newusers especially designed to answer these questions. Because of the size of the Usenet information space, several more specific new user groups exist. One example is the newsgroup comp.sys.mac.announce that, among other things, contains a monthly message about what subject topics belong in which Macintosh newsgroups. Another monthly message contains the answers to the most frequently asked questions in all Macintosh newsgroups (for sample messages see Chapters 13 and 14). These messages serve several purposes, but only to the extent that new users actually read these messages. The message about newsgroups helps stop users from posting to the wrong newsgroups and introduces new users to the general etiquette of the system. The message about frequently asked questions stops new users from irritating the general community and serves to introduce readers to the rules of the community.

3.5.1 Software Distribution

A sense of community is not the only reason that users actively participate in Usenet news. Usenet provides services that users have come to count on. For example, due to the global nature of its distribution channels, Usenet has become an important medium for the distribution of public domain and shareware software[1]. Many of the most useful and popular software packages have been available through Usenet for years. For example, the C language command line option parsing package *getopt* started off as a user contribution distributed over Usenet and has since become the defacto standard for option parsing. In addition, many commercial software developers make bug fixes and system enhancements available to Usenet readers through specialized newsgroups designated as software repositories. These newsgroups are specialized for the distribution of binary software images and source

1 Unlike public domain software, shareware costs money. However, users can evaluate software for free and pay only if the software proves useful.

code releases. When source code bugs are detected, users contribute patches that are often integrated into release software and distributed as patch files in the same newsgroups. Some software distributed in this manner becomes so popular with the community at large that it is purchased by commercial developers and redistributed through commercial channels. Still other shareware developers opt not to ever sell their software commercially. These developers commonly ask to be sent a postcard, or for software users to send a donation to their favorite charity.

The importance of software distribution through Usenet should not be underestimated. Binary distribution groups commonly appear at the top of the most popular newsgroup lists and many people read Usenet solely for the purpose of retrieving and discovering new pieces of software. One of the reasons for the popularity of Usenet software distribution is demonstrated by the proliferation of anti–virus software. Over the past two years the number of infection incidents has been declining [Norr 1992], in part due to the availability of free virus protection software. If everyone had to pay for virus protection, there would be many more viruses successfully spread throughout the computer industry.

3.5.2 Product Support & Evolution

Software development is enhanced by the existence of Usenet news. INFOSCOPE has been implemented in the MACINTOSH COMMON LISP (MCL) environment and much of the product support received during its implementation came from Usenet interactions. The willingness of the software developers to participate actively in these discussions allowed Usenet readers to have an active role in the development of the environment. As more readers began to contribute useful code Apple established a network accessible archive site for user contributions. Many months later when the official software was released, all of this user contributed software was included with the distribution on a CD–ROM.

The discussions of the development of MCL have also given way to the discussion of the next generation of object oriented languages currently under development at Apple Computers. As the level of discussion on this topic increased, and the lisp newsgroups were filling up with messages on this topic, Apple moved the discussion from newsgroups to mailing lists. Usenet was instrumental in the inception of the discussion and served a valuable role in the advertisement of the availability of the mailing list.

Finally, MCL is not the largest example of Usenet inspired software contributions. The amount of C language code (mostly UNIX utilities) available via Usenet is staggering. IBM (and compatible), Amiga, and Macintosh software also enjoy wide distribution across Usenet news.

3.5.3 Information Dissemination

Software distribution is not the only reason that users regularly participate in Usenet newsgroups. When a virus that attacked many UNIX machines was released on the Internet, Usenet newsgroups were instrumental in the quick dissemination of information on how to prevent and eradicate it. Subsequent to the virus incident,

newsgroups were lively with discussion on how to prevent similar attacks in the future. Situations like this show the importance of getting certain information in a timely manner.

In addition to the serious business of virus prevention, Usenet newsgroups disseminate information on a wide variety of topics. Newsgroups include those designated to discuss movie reviews, television shows, weather, research topics, political agendas, jokes, books, sports, cars and almost any other topic imaginable. The discussion of these topics contributes significantly to the feeling of community as participants in discussions find that they share common views with people they have never met. The willingness of people to spend time answering inquiries on so many topics makes Usenet a valuable source of current information.

3.5.4 Social Interaction

Usenet newsgroups are not used exclusively as a method to distribute software and disseminate recreational information. Newsgroups are used as a method of finding peoples' electronic addresses and are regularly used to announce that frequent contributors are moving from one job to another. One example of the usefulness of Usenet in this capacity is the yearly gatherings of readers that occur at several conferences. At the MacWorld conferences held in San Francisco and Boston readers of the Macintosh Usenet newsgroups regularly meet outside of the regular conference proceedings. These meetings are social events that bring together many people from everywhere in the United States. Many of these people have know each other for years, but only by their respective electronic mail addresses. They use the network to arrange meetings that lead to long term friendships and collaborations.

4. Motivation and Empirical Analysis

4.1 Information Overload

With the wide variety of information described above, the number of messages available in Usenet newsgroups is large. In fact, so many messages are available that no individual can possibly process all of it. This creates a situation called information overload that is discussed in this section along with other motivations for this research.

4.1.1 Definitions

Information overload generally refers to the problems created by the huge amount of information that is available to us as information consumers [Reddy 1983] but specific definitions vary. Psychological definitions refer to the drop in decision performance when decision makers are presented with more than 10 or so information items (cues) [Streufert 1973]. This definition is related to the limited short term memory capacity of the human brain. Other definitions emphasize overload effects rather than the causes. From that perspective information overload occurs whenever information processing demands on time and effort exceed the capacity of the information processor [Schick, Gordon, Haka 1990], independent of the number of items involved. This accounts for situations where the task is simple enough, or the items similar enough, that many items can be processed without overload effects. Even in those cases, however, the amount of information can become so staggering that overload effects occur. This is common in the accounting domain where accountants can deal with many numbers at once, but the huge amount of information in large spreadsheets leads to overload. It is important to recognize the contribution to overload that come from aspects other than the number of items. The cognitive effort required in processing information includes screening, comprehending, combining, evaluating, interpreting and using that information to make decisions [Schroder etal. 1967]. For this reason I will consider the more temporal definition as the practical definition for the INFOSCOPE project. If there is too much information to process then users experience information overload.

4.1.2 Sources

Sources of information overload also vary. Besides over 100 million volumes in the Library of Congress, over 70,000 scientific journals (108,600 journals in all fields) [Denning 1991] and thousands of periodicals, advances in microprocessor technology, communications networks, and computer systems will be unlocking access to new types of intelligent networks and information previously unavailable to computer users [Schwartz 1989]. With all this information available, we can compare the problem of finding interesting information to finding the proverbial needle in a haystack. No individual really wants to read all this information, even if that were possible. People are interested in learning about or investigating only certain areas of

personal interest. Uninteresting information (that which people do not wish to read) only confuses those tasks. For example, we often refer to unsolicited mail as *junk mail*. As methods of proliferating information advance, so does the amount of irrelevant information we must wade through to find the answers we need. This is especially true in a global bbs environment where users can post about anything they desire. However, the set of information each individual considers relevant varies according to personal preferences and tasks. So the stream of available information *must* be diverse. Unfortunately, as the diversity of information increases so do the deleterious effects of information overload [Iselin 1989]. INFOSCOPE addresses this issue by providing assistance in finding interesting information, within the diverse information space, on a personalized individual basis.

4.1.3 Effects of Information Overload

An important effect of information overload is demonstrated by an anecdote related in the book *The Sciences of the Artificial* [Simon 1981]. When the US State Department used slow teletype terminals for transferring messages, many important messages to Washington were seriously delayed during crisis situations. It was decided that replacing teletypes with faster line printers would allow the messages to get through on time. However, this solution led to the receipt of so many messages that the people responsible for reading them still couldn't get to some important ones in a timely manner. In a similar incident, air traffic control and telephone service in the New York area were disrupted when the backup power for the phone network failed. Several messages detailing the amount of backup power left were printed, but the few relevant lines of feedback were printed among hundreds of lines detailing call information. Had the appropriate messages been seen the entire incident could have been avoided. The real problem is finding a way of filtering out the important messages in times of emergency. This illuminates an important motivation for this research, namely addressing the fact that the limiting resource in message handling is not the availability of information, but the amount of time humans have available to attend to that information [Simon 1981]. This research effort hypothesizes that filtering messages can be an important tool in balancing this tradeoff since it concentrates user efforts on a select set of interesting messages.

While Usenet News represents only a small fraction of all the available information, it still represents a large enough information space to present its users with many of the effects associated with information overload. Empirical data, gathered through actual use of Usenet News, shows that even reading a few of the more popular newsgroups requires the perusal of megabytes of information each month (see Figure 4.1). With nearly 2000 newsgroups to choose from even the most prolific readers cannot read more than a small percentage. An empirical evaluation (carried out in the computer science department here at CU) showed that users are interested in reading about more topics, but the large volume of information they must wade through makes that impossible. Many of these responses were from readers who subscribe to only two or three newsgroups!

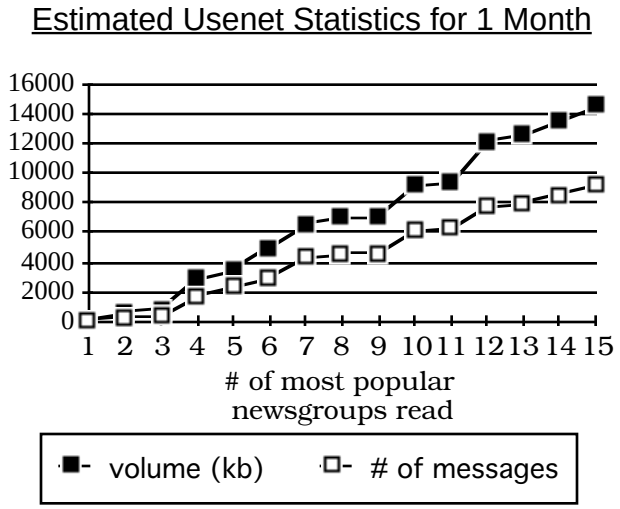


Figure 4.1 Usenet News Traffic

Usenet News with its 2000 or so newsgroups provides an excellent opportunity to study information overload. The line marked with white boxes shows that reading only 6 of the most popular newsgroups requires reading about 3000 messages in a typical month. The total information volume of these 3000 messages is 5000kb (the line marked with black boxes). This graph does not contain the popular source or binaries newsgroups that make the problem much worse.

Previous studies of information overload vary as widely in their results as in their domains. The domains of study include accounting [Schick, Gordon, Haka 1990], consumer marketing [Thukral 1983], television newsrooms [Whitney 1978], managerial decision making [Iselin 1989], survey design [Bergstrom, Stoll 1990], the display of books in libraries [Baker 1985], and bulletin board systems [Kelly 1987].

Two studies conducted in television newsrooms confirmed that the level of interest in information and the time devoted to that information were directly related [Whitney 1978]. The inverse of this conclusion has serious implications for the Usenet domain. As INFOSCOPE allows users to focus in on more and more interesting information, the time spent on those messages will reduce the total number of messages users will have the time to read. This makes it even more important that users are not presented uninteresting information that wastes time.

Another effect that has been associated with information overload is boredom [Klapp 1986]. Klapp found that boredom is directly related to the amount of noise found in information. For the purpose of this dissertation, uninteresting messages are considered noise. He points out that no independent analysis can judge what is considered noise by any person. Judgment of noise is essentially a personal task. While these results are not surprising, they lend credence to the INFOSCOPE approach of personalizing information spaces based on models of individual users. The amount of noise in a newsgroup can be determined only by message readers, not message senders or organizers of the default Usenet hierarchy.

Information overload effects the strategies employed by information consumers. One particular study took place in grocery stores and had consumers choose the *right* item from a number of similar items (the right item is the one the consumer repeatedly

chooses under non–overload conditions) [Thukral 1983]. In overload conditions where the number of items were high and the knowledge of each individual item was low, consumers consistently employed *negative bias choice heuristics* as a selection strategy. This means that instead of choosing the items based upon the positive characteristics of the desired item, undesirable items were first eliminated by concentrating on the negative aspects that made them easy to eliminate. Only after eliminating enough of the undesirable items to lower the levels of cognitive strain do consumers employ *positive bias choice heuristics.* This result effects the INFOSCOPE system in two ways. First, readers of high volume newsgroups will be choosing messages by eliminating the ones that are uninteresting. Second, once having defined virtual newsgroups users will be able to concentrate on the lower overload, positive choice heuristics. Defining good filters that create small concise groups should help this process.

In addition to effecting the choice of heuristics that are employed by people, information overload alters the effectiveness of those heuristics [Kilari 1989]. This can be seen in Usenet in the way readers choose messages. Almost all news reading systems display a list of subject lines for users to browse. Some actually display the entire header of a message. Users choose which messages to read based on these displays. However, Kilari showed that even if the whole header is displayed, one of the first effort reduction strategies users employ to reduce cognitive load is to reduce the number of attributes examined in each item. This means that in high load situations readers are likely to ignore most of the header just to be able to get through all of the messages. Kilari goes on to explain that this does not seem to happen in situations where the number of items is large, but they are all interesting. As long as sets of alternatives are filled mostly with interesting choices, users will tend to examine the items carefully. When there are too many uninteresting choices users will tend to look at fewer attributes. This may serve as a good argument for defining many filters, since filters never ignore the attributes they are assigned to scan. In addition, virtual newsgroups that contain mostly interesting messages will allow users to avoid the effects that uninteresting messages have on choice heuristics.

Information overload can effect the usage levels of a bulletin board system in general [Kelly 1987]. Even in small bbs's overload can occur. This has been shown to reduce the overall utilization and growth of the system. The more useful information people find, the more likely they are to return and use the system again. The reduction of overload effects on Usenet is crucial if the system is to continue expanding and including more users into the community. The diversity and amount of information available is at constant odds with this goal. Since overload problems cannot be solved by reducing the amount of uninteresting information posted, other mechanisms that do so must be found.

4.1.4 Strategies for Reducing Information Overload

As discussed in the previous section, it is important to lower the overload to "filtering levels" [Whitney 1978]. Overload must be lowered to levels that allow the use of positive choice heuristics. Those levels should be low enough, or contain a low enough percentage of uninteresting messages, that interesting messages can be

properly examined. Methods for approaching this problem have been classified into four general categories [Schick, Gordon, Haka 1990]:

- more efficient use of time:
  for example, overload can be reduced by creating user interfaces that allow users to properly organize and prioritize an information space

- fewer tasks to perform:
  for example, overload can be reduced by carefully analyzing the proper distribution of responsibility between system and user [Henninger 1990]

- more time available:
  for example, overload can be reduced by wasting less time scanning uninteresting messages

- expand size of workforce:
  for example, overload can be reduced by hiring a secretary (or building a computer system like INFOSCOPE) to prioritize messages

The INFOSCOPE system approaches the overload problem by considering each of these strategies. Users are able to more efficiently use their time by using INFOSCOPE's graphical interface characteristics to browse the information space. By delegating as much responsibility to the system as possible, INFOSCOPE reduces the amount of effort users must expend to a minimum. However, it should be noted that INFOSCOPE does create the extra burden of dealing with suggestions (see Chapter 7). More time is made available to INFOSCOPE users through the development of personalized structure that eliminates many uninteresting items. This is an evolutionary process so at first not much time savings is evident, but as filters are defined the user is able to save more time. The size of the workforce is effectively expanded in INFOSCOPE by the introduction of agents. These agents act like other workers by scanning your messages and helping you prioritize them. The major issues raised by this approach include keeping the agents from being too intrusive (or creating too many suggestions), and making the agents flexible enough to adjust to the preferences of different users.

4.2 The Vocabulary Problem

In addition to the problem of information overload, the vocabulary problem [Furnas et al. 1987] prevents people from efficiently processing information. Terms used by one individual to classify an information item may not serve as appropriate retrieval cues for others in different situations or task domains. Studies have shown that retrieval performance is reduced anywhere from 25% to 50% in systems organized by someone other than the information retriever [Broadbent 1978; Piekara, Strube 1987]. In addition, recall of descriptors is significantly higher when those descriptors were generated by the test subject rather than simply encountered through reading [Bobrow, Bower 1969]. These problems manifest themselves quite prominently in the use of the newsgroup hierarchy. When users post to a particular newsgroup they are necessarily imposing their own semantic interpretation on the message by placing it in a specific part of the newsgroup hierarchy. Whoever originally defines the newsgroup names similarly imposes personal bias on the Usenet structure, making

everything clear personally but creating conceptual and vocabulary mismatches for others [Winograd, Flores 1986]. The problem is that not everyone understands each classification in the hierarchy to contain the same set of message topics. For example, a person looking for information about the UNIX visual text editor (vi) mistakenly looked in the comp.lang.visual newsgroup. The problem is that this newsgroup deals with issues related to visual programming techniques, not the UNIX visual editor. One approach to reducing this problem is allowing the hierarchy to evolve. This has been implemented by having the entire network community vote on new newsgroups, but the process is slow and requires the agreement of significant numbers of people. So changes often get proposed several times before enough users are convinced or interested enough to cast votes in favor. INFOSCOPE allows users to define their own refinements of the hierarchy without waiting for, or being subjected to, the compromises of the Usenet community at large.

This occurred with the newsgroup comp.sys.mac from 1989 through the middle of 1990. Messages flowed back and forth in the group supporting both splitting the group into several pieces and keeping it a single newsgroup. Finally, the group was split into subgroups including comp.sys.mac.comm (for communications programs), comp.sys.mac.system (for operating system related messages), comp.sys.mac.apps (for messages about application programs in general) and of course the catch–all group comp.sys.mac.misc (for whatever you can't find a place for in other groups). Problems still remain, however, for those individuals who do not agree with accepted changes. They must still access information through other people's conceptualization of its structure. For example, where does one find a message about a communications program that crashes the system when run with a particular word processor? Is this an operating system problem, or a communications problem, or a word processing problem? Using INFOSCOPE, users can find information about their specific programs, regardless of where those messages are actually located. Filters can easily scan potential sources for occurrences of relevant keywords.

Another example of how the INFOSCOPE approach can solve certain problems involves discussions about the object oriented programming language CLOS (Common Lisp Object System). During the development of the language specification, discussions about relevant issues appeared in both a mailing list[2] and the newsgroup comp.lang.lisp (discussions about the lisp language). When I first became interested in CLOS as an object oriented language in INFOSCOPE, I found myself having to peruse many uninteresting messages in the comp.lang.lisp newsgroup. The amount of irrelevant information in the mailing list was significant because I was not interested in the low level implementation details of particular CLOS functions. I was interested in seeing information about the general concepts involved in the CLOS language. Using INFOSCOPE I defined a virtual newsgroup called comp.lang.lisp.clos into which I filtered any messages containing references to PCL or CLOS. I was then able to read interesting messages without having to manually recognize and ignore uninteresting messages. Well over a year later the Usenet population voted to create the official newsgroup comp.lang.clos for the very same

2 The mailing list was actually about issues involving the evolution of PCL (Portable CommonLoops) into an implementation of the CLOS specification. Many important CLOS issues were commonly discussed in this forum.

purpose. I simply added the new official newsgroup to my existing filter and continued to read messages as I had been doing for some time.

Some small empirical studies have provided further evidence that there is a serious vocabulary problem in the Usenet domain. In the first study I found an announcement for a conference in the area of cognitive science and human–computer interaction. I then asked a group of HCI researchers to (a) pick the one newsgroup into which that announcement should be posted, and to (b) choose several other newsgroups that might be appropriate for that message. Of the ten participants in this study only 3 chose to post the message to the newsgroup news.announce.conferences (the culture's official repository for any conference announcements). Others split the posting into various special interest newsgroups covering several topics. Overall, there was very little agreement as to where that particular posting belonged. The problem should be even more dramatic when the message really doesn't have a specific newsgroup in which it belongs. A full description of the required posting and the specific results can be found at the end of this proposal (see Chapter 15). One problem here may be that people do not ever read the messages that define where certain types of postings belong.

The second study to indicate a serious vocabulary problem in Usenet News investigated the structures users create when saving messages they deem interesting to their personal information spaces. Even within a close set of researchers who investigate similar issues (i.e., the HCC group here at CU), there is very little agreement about classifications. Some users see News messages as an entity unto themselves, saving all messages from any newsgroups into the same file (perhaps called "News"). Others create detailed hierarchies of files and directories to contain these saved messages. Even when the structure seems to match another user's structure, the names attached to these files or the contents of similarly named files tend not to match each other. Users don't attach the same semantic meaning to the terms they use for classification.

Finally, graphical linguistic features of messages may have significant vocabulary problem effects. An example is the commonly used smiley face (:->). A message reader who does not understand the meaning of this feature (i.e., the indication of a joke or "tongue–in–cheek" commentary) may easily mistake a flip remark for a serious position. This has indeed led to many arguments in newsgroups. The only solution to this problem is making all users aware of the Usenet culture and it is not clear how much INFOSCOPE helps here, but the fact that this causes interpretation mismatches is evident when reading newsgroups.

Creating virtual newsgroups can reduce the effects of the vocabulary problem. This is accomplished by defining filters that map potential semantic mismatches into appropriate virtual newsgroups. For example, if readers are not sure where someone might post a message about conference announcements, they can define a filter to look in several potential newsgroups. Any conference announcement posted to any of these newsgroups is collected into a single location, perhaps a virtual newsgroup called conference.announcements or whatever name makes sense to the individual.

4.3 The User Interface to News

Several interfaces to the news information space have been implemented. Most of them have been text based and designed to operate on conventional tty type terminals. The most common of these interfaces is called RN. This system presents users with a succession of text lines containing the names of newsgroups and the number of unread messages in that group (see Figure 4.2). The software keeps an initialization file (.newsrc) for tracking the newsgroups subscribed to by the user. A user subscribes by requesting messages from that group. This file also determines the order in which newsgroups are presented but the system comes with no tools for managing this file. To override the default order users must either edit the initialization file or ask the software to go directly to a specified group. While this type of an interface might be fine for a user who is both experienced with the software and familiar with the organization of the information space, observations and discussions with casual users indicated that they have never really liked these interfaces and several attempts to improve on this paradigm have been implemented (see Chapter 6.1).

The RN Interface to Newsgroups

```
***   1 unread article  in alt.hypertext--read now? [ynq]
***  13 unread articles in alt.sources--read now? [ynq]
***   1 unread article  in boulder.general--read now? [ynq]
***   2 unread articles in co.general--read now? [ynq]
***  23 unread articles in comp.ai--read now? [ynq]
***  14 unread articles in comp.binaries.mac--read now? [ynq]
***   9 unread articles in comp.cog-eng--read now? [ynq]
***  99 unread articles in comp.lang.lisp--read now? [ynq]
***   1 unread article  in comp.mail.headers--read now? [ynq]
***   6 unread articles in comp.newprod--read now? [ynq]
***  11 unread articles in comp.sources.games--read now? [ynq]
***  30 unread articles in comp.sources.misc--read now? [ynq]
***   1 unread article  in comp.sources.unix--read now? [ynq]
***   1 unread article  in comp.sys.mac.digest--read now? [ynq]
*** 289 unread articles in comp.sys.mac--read now? [ynq]
```

Figure 4.2 Text Based Newsgroup Interfaces

Text based interfaces do not display the relationships between newsgroups in the information space. Browsing in this type on environment can be an arduous task.

4.4 Support for Conversation

Another aspect of the RN software that confounds many novice users is the lack of structure imposed on logical collections of messages. Conversations between people are often more than “one–shot” dialogs. When several users post a response to a particular message, those messages may each spawn sub–conversations themselves. These sub–conversations are often called *threads*. RN displays messages in the chronological order in which they arrive at an individual's computer. Although the system provides a command for scanning to the next message in the complete conversation, the structure of individual threads is lost in the chronological ordering. Because of this, senders often include whole conversations as preambles to their own contributions to those conversations. This leads to wasted effort by the reader as well as wasted bandwidth on the networks. In fact, due to the way messages are propa-

gated throughout the network, situations can actually arise where a response to a message is displayed prior to the original message. INFOSCOPE attempts to avoid these problems by organizing messages into cohesive conversations. Responses to messages are displayed hierarchically as children of the messages to which they are responses. Responses to responses are similarly displayed as children of the initial response.

5. Conceptual Background

The INFOSCOPE project is not an isolated attempt to solve problems in the information access domain. It is only one part of an overall attempt to develop the technologies necessary for the realization of personalized information environments. In order for these systems to be usable, they must include some amount of knowledge that can be used to shift much of the information processing burden from the individual to the system [Fischer 1990; Henninger 1990]. An integrated system must help deal with information throughout its lifetime, instead of just at one point in its life cycle as isolated systems tend to do.

5.1 An Information Life Cycle

Like many artifacts in our lives, the messages we send and receive have their own life cycle. When developing systems to help with messages we must recognize the processes that a message undergoes in traveling from step to step in its life cycle. For example, as illustrated in Figure 5.1, when related messages traverse the arc between *read time* (when messages are read by someone) and *send time* (when messages are created) a *conversation* is often the result. This is what happens when someone replies to a message. However, only some messages are interesting enough to keep, and those messages traverse the arc between read time and *storage time* (when messages are saved for later retrieval). Saving messages is one of the operations INFOSCOPE uses to help determine which message topics are interesting. Saving message indicates some level of interest to the person who saved it.

Once a message is stored for future reference this is not the end of its life cycle. When people have a question they can access their personal information store to find relevant information. This process of taking a message from *question time* to read time is called *retrieval*, and is addressed by the HELGON system [Fischer, Nieper-Lemke 1989], as well as many other retrieval systems. A fundamental difference between read time and question time is that at read time the task is to find interesting information, while at question time there is a more specific task at hand [Fischer, Nakakoji 1991], namely finding the answer to whatever question requires the information search. In fact, it is in anticipation of this question time task that users go to the trouble of storing information in their personal environments. When a user finds a relevant piece of information it can also be modified. This is often the case when the

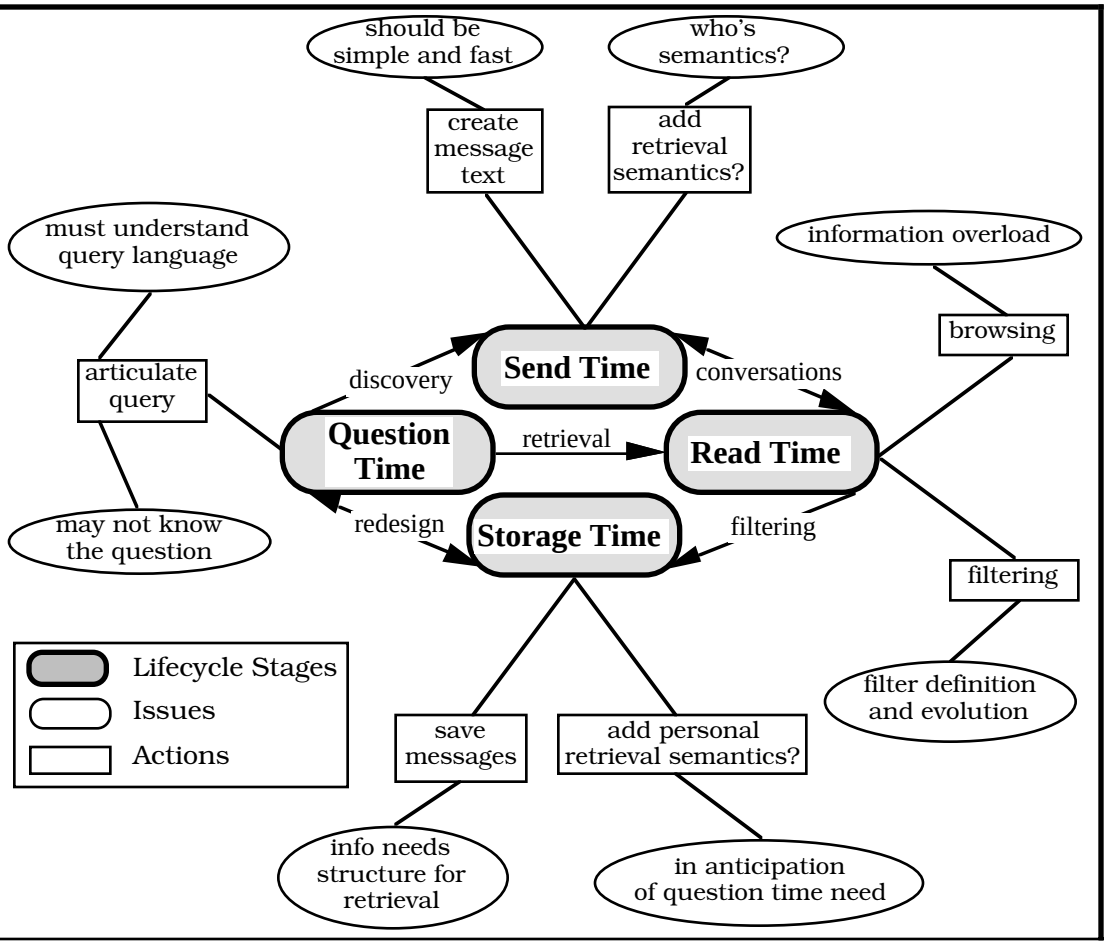


Figure 5.1 An Information Life Cycle

There are four basic stages in the life of a message. Within each stage there are several tasks and decisions to be made. The arrows between stages in this diagram represent the processes involved in transferring a message from one stage to the next.

A message is created at *send time* and either addressed to a set of individuals or posted to some sort of bulletin board system. At *read time*, people looking for information or answering e–mail must decide which of the newsgroups and messages are worth reading. Sometimes, when a message is of particular interest it is stored by the user for later reference. At this point (*storage time*) the user adds some sort of semantics to the message, ranging from simply specifying a filename to adding keywords and classifications. Lastly, stored messages are retrieved at *question time* when a question triggers their need.

content of particular messages contains artifacts like source code. The process of modification and subsequent storage back into the personal environment as a new object can be referred to as *reuse and redesign* [Fischer, Henninger, Redmiles 1990].

Lastly, there is the situation where the users have a question, but have not previously stored any relevant information in their personal environments. In this case, users engage in a process of *discovery* [Schwartz 1989]. In this context discovery means that a new message is posted to the outside world as a request for augmentation of one's own environment. Since discovery may involve interaction with many agents (here the other users of Usenet News), it is unlikely that any two identical queries will result in the same set of answers. The same query to the same database by several individuals should always yield the same results in a retrieval situation. This makes the process of discovery quite different from that of retrieval. Another distinguishing factor between retrieval and discovery, is that discovery involves a search for previously unknown information, while retrieval involves the search for previously stored information. In order to retrieve something, you must know where it is. For example, over the years I have saved many messages about problems users encounter on the Macintosh computer. When I have a problem I often refer to this set of messages to find a solution. This is a retrieval process. More often than not, however, I am encountering a new problem and I haven't already sorted the answer in my personal information space. In this case I post a message to a newsgroup asking for help. This is a discovery process. I have no idea what I will get back, and if I get nothing I can try again. If a different set of people read the second posting, I may very well receive an answer.

5.1.1 Send Time

The whole concept of the information life cycle cannot be expressed through the arcs that transfer messages from one step to another. The four distinct processing steps each represent several actions and issues with which users must deal (see Figure 5.1). The first of these is *send time* processing. Send time processing is the work users must do to send a message. This processing ranges from simply specifying a destination and subject (as in UNIX e–mail), to coarse grained categorization (as in Usenet newsgroups), to fine grained structuring of the semantics of the message content (as in the LENS system discussed in section 6.2.1). It is important to support send time processing without burdening message senders with tasks that do not benefit them.

5.1.2 Read Time

After send time processing the next step is *read time* processing. Read time processing is the work users must do to find and read the interesting information from within the complex information space. Since the reader is typically a different person than the sender, this involves a range of activities from simply browsing the information space (like the UNIX RN news reader), to writing specific rules to filter out information that already includes sophisticated semantics (like the LENS system). At read time, users perceive more benefit from categorizing information because they are the direct beneficiaries of that effort. This has limitations though. An empirical study of UNIX file systems showed that users often cannot recall the meanings of their

classifications because the rationale for the classifications are not explicit. If users do not remember where they put a piece of information, or what they called it, finding this information can be a difficult task. These are issues that should be addressed at storage time.

5.1.3 Storage Time

Once a message has been read, a decision must be made as to whether the message should be saved for future retrieval. If the decision is for saving the information, the user then has an additional opportunity to add retrieval semantics to the message. We call this storage time processing. This ranges from simply saving all messages to a flat file or mbox (like many people use UNIX e–mail), to adding coarse grained semantics by saving to a folder structure (like some people use UNIX e–mail), to utilizing fine grained semantics and saving to a public area which others can peruse using rules (like the LENS system). Any semantics added at storage time (file names, directory paths, additional keywords…) can be used by a retrieval system to help users find their information later.

5.1.4 Question Time

Finally, there is the processing which must take place when the information is needed later. This is the information retrieval problem as discussed in the literature and we call this question time processing. Question time processing ranges from simply searching for text strings, to utilizing coarse grained semantics (like the UNIX e–mail folders), to utilizing fine grained semantics in the information (like the HELGON or LENS systems). In addition, systems like HELGON [Fischer, Nieper-Lemke 1989] or CODEFINDER [Henninger 1990] that do not rely on users retrieving with exact terms help overcome the problem described in the UNIX file system example above.

5.2 Situation and System Models

In addition to the information life cycle, an analysis of the situation and system models [Dijk, Kintsch 1983] plays an important role in the conceptual framework of the INFOSCOPE project. These concepts originated in the domain of text comprehension, but transfer well into the domain of utilizing computer systems. People comprehend text, state their desires and understand interactions in the terms of their current task, context, past experiences and view of the world. This is called the *situation model*. Systems (computers and other systems/devices), however, require interactions with people that are organized around someone else's view of the world (the *system model*). For example, a person only drives cars equipped with automatic transmissions. If that person gets into a car with a manual transmission for the first time, they will not know how to operate the vehicle. The problem is that the person's situation model, driving with automatic transmission, differs significantly from the vehicle's system model, that requires knowledge of driving with a clutch mechanism. The larger the gap between situation and system models, the harder time people have bridging that gap. This means that the person who at least knows something about how a clutch works has a better chance in moving the car than the person who doesn't.

In our research group we have investigated various methods for bridging the gap between situation and system models. Several approaches offer the user help in translating their situation model into the appropriate system model (see Figure 5.2). Another approach (taken by virtually all commercial software products) is training users to express themselves directly in the appropriate system model. This requires users to read and comprehend long and complicated manuals. The drawback of these approaches is that at some point the user must map from their personal situation model to a more global system model, a task that consumes valuable cognitive resources. An additional approach (investigated through INFOSCOPE) is to develop systems that learn about users to allow the expression of structure in the user's own situation model. The system model presented by INFOSCOPE is refined over time, through the development of a persistent user model, until its system model closely matches the situation model of the user. As users read messages over time, INFOSCOPE constructs a model of user interests. These models are used to suggest modifications to the system model that closely match the semantics users are familiar with. As time proceeds and the user models improve the system model presented by INFOSCOPE contains more and more of the terms individual users find relevant (i.e., their situation models). As these modifications are made the user is required to map less and less into a system model. A central research issue for this project is determining to what extent this mechanism for system model modification is useful in the Usenet News domain. The hypothesis is that by restructuring the information space, users will be able to find more interesting messages. This in turn should allow them to broaden the scope of newsgroups they read by requiring them to spend less time browsing uninteresting messages.

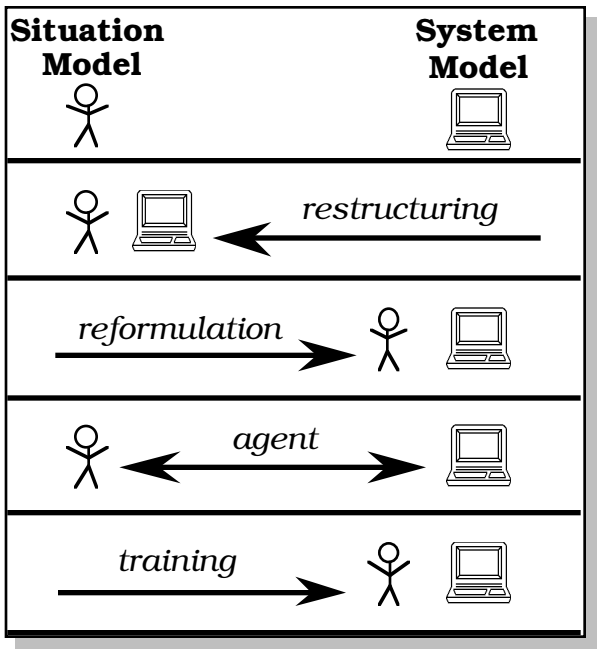


Figure 5.2 Different Approaches in Relating Situation and System Models

The situation model / system model theory allows for analysis of the gap between what users know, and what they must know to use and understand systems. By better understanding the various ways to bridge this gap I hope to gain a better understanding of the actual contents of a situation model.

- The first row illustrates the situation where there is no support for bridging the gap.
- The second row shows the approach in which a new system model is constructed that is closer to an individual's situation model and hence easier to understand. This approach is pursued with the INFOSCOPE system.
- The third row illustrates the possibility of making the system model more transparent; allowing users to express their situation model incrementally within the system model.
- The fourth row shows how an agent can help translate a query from the situation into the system model.
- The last row illustrates the training approach, where users are trained to express themselves in the system model. This is not appropriate for situations where the tasks/interests/queries change since these changes cannot be trained. This is why restructuring and reformulation must augment training.

5.2.1 The Gulfs of Execution and Evaluation

The notions of situation and system models are not the only attempt to explain the difficulties encountered by users of computer systems (or any *designed* artifact). Norman discusses a similar perspective on this problem and refers to it as the gulfs of execution and evaluation [Norman 1986]. In his analysis, two gulfs exist between a user's goals and the physical system in which those goals must be expressed. The first of these gulfs is called the gulf of execution because it involves the translations necessary to go from one's goals to the physical system used to execute those goals. The gulf of execution is bridged by making the system match the user's view of the domain as closely as possible.

The second gulf is the gulf of evaluation. This gulf corresponds to the problems of mapping from the physical system that has completed some portion of the desired task to the terms and expressions of the goal as viewed by the user. This gulf often

causes problems since the representations that constitute an answer on the computer screen are not readily interpretable by users. The gulf of evaluation is bridged by ensuring that the output of the system presents an understandable conceptual model to the user. INFOSCOPE shows that bridging these gaps is not the only way to span them. INFOSCOPE instead spans the gulfs by actually bringing the physical system closer to the users goals. The physical system is eventually represented in terms gathered in the user models.

5.2.2 The External–Internal Task Mapping Analysis

Another analysis of the problems users encounter when attempting to complete some task on a computer system is Moran's external–internal task mapping analysis [Moran 1983]. In this model the computer system is viewed as "a conceptual world into itself — it represents conceptual objects and carries out conceptual actions on those objects." Again, this analysis points out that the major problem is that computer systems are designed, and in the process of design certain conceptual mismatches arise between the system's internal representations and the external reality of tasks and their specifications in the domain of interest to the computer user. The external–internal task mapping model is a design model that stresses the structure of the external task as opposed to its requirements. So one of the postulates of this model is that systems must provide users a method for getting their task "into the system" by reformulating the specification of the task until is understood by the system (see Figure 5.2)

5.3 The Role of Structure

Information spaces can be organized with various degrees of structure. At one extreme there is no structure at all. In this case, searches are little more than undirected browsing. At the other extreme very rigid structure dominates interaction with the information space. Databases employ this amount of structure and require users to know specifics about it in order to access information. This research explores the possibility that an evolving flexible structure is often more appropriate. In the INFOSCOPE system there are two types of structure to discuss. The first of these is the *a priori structure* that defines the Usenet hierarchy of newsgroups. The second is the *virtual structure* that INFOSCOPE allows users to create and that serves to give the information space structure its flexibility.

5.3.1 A Priori Structure

The original set of Usenet newsgroups was defined by the administrators of a group of machines called *the backbone*. These are the machines that bear the largest burden of the communication (message transfer) costs associated with feeding the many news distribution sites. The overwhelming majority of existing newsgroups were part of that original newsgroup organization. Subsequent to creation of these groups, users of Usenet have agreed, through protracted discussion, that extensions to the hierarchy would reduce clutter and better organize the information space. As previously described, however, this process takes a long time and there is no possible way for a

group of users as large as Usenet participants[3] to find a solution that fits every user's semantic interpretation of the newsgroup's contents. As soon as a relatively small majority of users interested in a proposed change desire that change, the newsgroup hierarchy is extended or modified.

A structure seems a priori from the perspective of any user that did not participate in its definition. For example, some of the structure of a newspaper is a priori from the perspective of a reader who is new to a city. In one city there may be one set of sections with associated names, and that may be quite different in another city. Each morning a group of editors meets and decides which articles to include and which sections of the paper to put them in. If the classifications the editors choose (the *system model* from the reader's perspective), do not match the way an individual reader would classify those same articles (the *situation model*), it can be difficult to find an article since it may not be clear what section to look in (see Chapter 5.2). An article on a proposed baseball stadium could be placed in the front section (if it is considered very important or controversial), the sports section (since those are generally the readers interested in sports issues), the local news section (since it is of local interest), or even the business section (if the article deals with the tax implications of building the stadium). Since the structure of a newspaper rarely changes, articles about every conceivable topic must be made to fit in that static, a priori, structure. This is a clear manifestation of the vocabulary problem (see Chapter 4.2), that is unavoidable in information spaces utilizing a priori structure.

5.3.2 Virtual Structure

Virtual structure is an approach to the problems caused by a priori structure. From the theoretical perspective, virtual structure helps span the gap between situation models and system models. The problems created by this gap relate to the fact that the system models offer qualitatively different semantics than users of systems experience in the daily life. By redefining existing structure based upon personal semantics, system models are brought closer to situation models. In the context of this particular research project, virtual structure allows users to manipulate newsgroups that contain collections of related messages. The important point is that the definition of *related* comes from user semantics, not the semantics chosen by the backbone machines. As virtual structure evolves, it may be possible to discontinue interaction with a priori structures all together. An example of virtual structure is the comp.lang.lisp.clos newsgroup that I described creating in Chapter 4.2.

5.3.3 Issues Related to Structure

An important characteristic of structure is that its modification requires cognitive effort. This means there are associated costs and benefits which people can analyze to decide if performing a given task is *worth the effort*. Research in computer supported cooperative work has shown that people are reluctant to perform tasks for which there is little perceived benefit to the performer [Grudin 1989]. So, getting users to modify structure means they must feel the modification will benefit them later. If this is not

---

3 There is no way to get an accurate count of all users who read news, but participation estimates routinely exceed one million users.

the case, the result is no modification or sub–optimal modification of the structure in question.

In addition, when discussing retrieval systems that utilize a rich structure it is important to ask, "*where does this structure come from*?" In the cases of most databases and the predefined Usenet hierarchy, system administrators define the structure. Users of these systems may take part in the addition of more structure, but the original conceptual layout of the structure remains throughout its lifetime. This situation is a major source of the vocabulary problem in utilizing the News hierarchy. Considering other messaging systems, like THE INFORMATION LENS (see Chapter 6.1), that require even more structure we might ask *"how much structure is the proper amount of structure?"* Requiring too much structure can overburden users while providing little advantage over less structured systems. Another goal of this research effort has been to explore how message handling systems may be designed to automate the generation of this required structure by making restructuring a normal and beneficial part of the message handling process. Studies of the INFORMATION LENS system have shown that users are willing to make the definition of information filters part of their regular message handling activities [Mackay 1990b]. INFOSCOPE extends this research by showing that systems can play an active role in assisting users with creating these filters (see Chapter 9).

There is no point in having a rich structure if it is not used to browse the information space it represents. To do this, users must have a clear conceptual understanding of the structure as represented by the computer system. Because the News hierarchy is predefined, there is an a priori mapping of semantics to newsgroup names that may or may not match the semantic interpretations of these mappings by any individual user. Without a clear understanding, the structure can do more harm than good. For example, an empirical analysis carried out on users of the HELGON system indicated that users were confused when the relationships between nodes were unclear due to the vocabulary that was used [Fischer, Nieper-Lemke 1989]. As a participant in this study, I personally was confused by the terms *proceedings*, *in–proceedings*, and *article*. When asked to find an article that appeared in a conference proceedings I was not sure where to look. This made the assigned retrieval task more difficult. INFOSCOPE addresses this problem by adding structure at storage time that is directly based on the structure added by the individual users. This personal structure can be used to augment a priori classifications. If I save a message that was filtered into a virtual group with the term *articles* in the name, that term is save along with the message and can be used to retrieve it in the future.

As the previous discussions indicate, information access systems are faced with the dilemma that:

- users need rich structure (hierarchies…) in information spaces so that a clear understanding of the conceptual structure is maintained through comprehensible organization,
- many sources of information do not exhibit rich structure,
- a priori structure is insufficient due to mismatches between intended and interpreted semantics,

- users are willing to add structure themselves only if there is a perceived benefit from adding it.

5.4 Personalization

This dilemma leads to a need for personalization of the information space. The methods for doing this however, must maintain that perceived benefit. The key to doing this lies in allowing users to add structure at read time as opposed to requiring others to add it at send time. In this way users benefit directly from the effort expended in modifying their personal view of the information space.

A commercial personal information manager called AGENDA [Kaplan etal 1990] similarly recognizes the need for information to be personalized by the user of the information, not necessarily the originator of the information. It also recognizes the need for flexible, evolving structure in these personalized environments. In this system users create personalized *views* of their data and AGENDA helps to automatically classify new items into those views. A major difference between AGENDA and INFOSCOPE, however, is that in the Usenet News domain users must find information in a structure previously defined by someone else. AGENDA is designed primarily for classifying personal information that is created by its user, not found.

There are other indications of the need for personalization. Since INFOSCOPE must allow the expression of structure in a user's own situation model, a single system model may not be adequate. This research hypothesizes that many system models that closely match individual situation models are more appropriate. Anticipation of every conceivable situation model is impossible since situation models are highly dynamic in nature. Therefore, rather than force the designer to attempt this impossible design task, it may be better to allow the user to define his own system model based upon his own situation models. In INFOSCOPE this is accomplished through the definition of *virtual newsgroups*. The system and users define these virtual structures to personalize the displayed system model (e.g., the current hierarchy). Using this facility, the users in the earlier example who needed to read about CLOS but instead had to read commonloops and lisp newsgroups can define a new newsgroup called comp.lang.lisp.clos (or whatever they choose as newsgroup names). A filter specifies that certain messages from other groups in question should be placed in the new group. Since the new group only exists within the personal environment of the defining user, and not in the public version of the Usenet structure, it is called a virtual newsgroup.

5.4.1 Filtering

As discussed before, the newsgroups in Usenet News are arranged in a general classification hierarchy. As demonstrated by experiments with the HELGON system [Fischer, Nieper-Lemke 1989], representing the intermediate nodes of a hierarchy helps make the system model more transparent and therefore, easier for the user to understand. For example, showing the entire pathname of a file in a UNIX environment usually gives one more information than only knowing the name of the file. In the INFOSCOPE interface not only are newsgroups represented, but so are the entire virtual and predefined hierarchical structures. Hence, not all nodes in the screen

display represent newsgroups. Some nodes in the INFOSCOPE display contain messages and are therefore newsgroups. Others contain additional nodes that refine the hierarchy. In this way, INFOSCOPE nodes are similar to UNIX files (which also serve as directories) or MACINTOSH folders/documents.

Unfortunately, system model personalization through the modification of the newsgroup hierarchy is difficult. Specifically, my own experience with defining virtual newsgroups shows that maintenance of these structures requires significant effort. In order to define good filters, users must evolve the filters as their interests change over time. Stale keywords that are no longer of interest must be removed and new interests must be reflected [Mackay 1990b]. What is needed is a suite of tools that make the task easier to accomplish. To do this in a computer system we need to model the user based on certain concepts of how users act, and we need to develop methods to analyze these models for structure creation and modification. In INFOSCOPE, these methods are encapsulated by agents (see Chapter 8). Another reason that users require assistance in defining filters is that people are poor intuitive predictors [Kahneman, Tversky 1973]. They tend to be insensitive to statistical evidence or prior probability. They ignore the past in favor of the current evidence. As pointed out above, ignoring the past leads to the definition of sub–optimal filters.

Aside from the complicated task of changing the input language of a system through the evolution of its system model we must pay close attention to the system's output language as well. In order for the situation/system model gap to be completely spanned, the personalization of the input language must become apparent in the interfaces output language. In the INFOSCOPE system this means creating different representations for the virtual structures that users and agents create and modify over time. INFOSCOPE uses a modified version of the TRISTAN graph display system to accomplish this new representation [Nieper 1985]. Nodes that represent refinements of the classification hierarchy are displayed in square nodes while actual newsgroups are displayed as oval nodes. Virtual newsgroups are displayed as highlighted oval nodes (see Figure 1.1).

5.4.2 Configurations

The filtering of relevant messages into personalized repositories is not the only method of reducing the information overload in Usenet news. INFOSCOPE users have indicated that they have very little idea of what newsgroups are available because of the large number of classifications. INFOSCOPE approaches this problem by providing users a mechanism for defining personalized views into the information space. These views are called configurations. Using these configurations users choose views of information that group related information sources together. For example, users can define a different configuration for browsing information relevant to work than they define for browsing recreational information. This helps reduce screen clutter and is especially effective for smaller screens where the users must scroll to view the entire displayed space.

5.5 User Modeling

User modeling is a wide research area of its own that is not examined deeply in this dissertation. However, INFOSCOPE utilizes proven techniques for user modeling in order to keep track of user interests. For a user model to be effective in carrying out this task it must have at least three characteristics [Mastaglio 1990]. User models must be dynamic; they must change over time. User models must be persistent; they must be retained between user sessions and reused by the system. Finally, user models must be idiosyncratic; they must be different for each individual user. INFOSCOPE user models satisfy all three of these criteria (see Chapter 8). Regardless of which of these characteristics a user model has, there are several method of getting information into a user model.

5.5.1 Implicit User Model Acquisition

Implicit user model acquisition is a matter of designing systems that observe user actions. These observations may be used to classify users into stereotypical groups or to supply rule systems with the necessary information to make relevant deductions or implications [Mastaglio 1990]. Stereotyping can be an effective method for leveraging a limited amount of information. By grouping individuals into stereotypes systems can use information about the entire group to make decisions about individuals within the group. Stereotyping does, however, have its limitations [Kass, Finin 1987]. INFOSCOPE uses implicit acquisition to recognize patterns in user interest that message readers may not be aware of. This type of implicit acquisition is effective in tracking long term interests because it operates constantly without being intrusive. However, implicit acquisition can only gather information about users that is demonstrated through their use of the system and cannot capture potentially relevant information about users that it cannot observe. INFOSCOPE does not attempt to stereotype users but there is no reason that this functionality could not be added in future versions. Evidence shows that users within a group are willing to share filters with their colleagues [Mackay 1990a] and identifying users that are likely to share interests can be used to help present them with catalogues of relevant filters designed by other users [Nakakoji, Fischer 1990].

5.5.2 Explicit User Model Acquisition

Another method that has been successfully used to acquire user models is explicit user model acquisition. In this method users are asked to complete questionnaires and that information is entered directly into the user model. This method is especially useful in acquiring initial user models from new users. In INFOSCOPE, users supply the location of the data files used by their past news reader and that information is used to indicate what newsgroups users are already browsing. In addition, INFOSCOPE, provides users with methods to explicitly inform the system that specific messages or terms are interesting. These explicitly acquired items are weighted more heavily than implicitly acquired information so that conclusions based on it are made faster. Other systems that filter information implement explicit user models that require users to supply the system with a set of messages that are considered representative of their information interests [Wyle, Frei 1989]. The drawback of these approaches is that self evaluations do not account for information interests that are not

explicitly recognized by users at the time of the acquisition. In order to avoid this problem, systems must either observe users implicitly, or frequently require users to update their self–evaluations.

5.5.3 Statistical User Model Acquisition

Statistical user model acquisition is an extension of implicit acquisition. The system observes user actions and collects this information for the purpose of statistical calculations. Information is collected until threshold levels are reached and actions are taken. This technique was successfully used in the ACTIVIST system to trigger an active help system [Fischer, Lemke, Schwab 1984]. INFOSCOPE collects raw data over long periods of time that is passed onto agents. Agents massage the data into patterns that are then subjected to statistical analysis.

5.6 Adaptive and Adaptable Systems

Adaptive systems are those that can autonomously modify a system while adaptable systems are those that allow users to modify them [Girgensohn 1992]. Each of these types of systems has their advantages and disadvantages. While adaptive systems are able to adjust themselves to the needs of individual users, the users loose control over the system. An example of adaptive systems can be found in [Fischer, Lemke, Schwab 1985]. Adaptable systems allow users to maintain needed control but require users to expend significant amounts of effort to carry out their adaptations [Fischer 1992]. Examples of adaptable systems are EMACS [Stallman 1981], NOTECARDS [Trigg, Moran, Halasz 1987], and THE INFORMATION LENS [Malone, Grant, Turbak 1986].

This research attempts to address the problems of both adaptable and adaptive systems by combining adaptive and adaptable components into one system. While THE INFORMATION LENS allows users to manually define information filters, INFOSCOPE takes the additional step of supporting adaptivity with rule–based assistance in creating these filters. Agents monitor filters to ensure that they more accurately represent actual patterns of user interest. INFOSCOPE is adaptive and can respond to interest patterns even when users are unaware of them. By combining these capabilities INFOSCOPE operates in the mode of shared decision making (see Figure 5.3). The combination of methodologies is more powerful than either one alone because the user and system can carry out the task for which they are best suited [Henninger 1990].

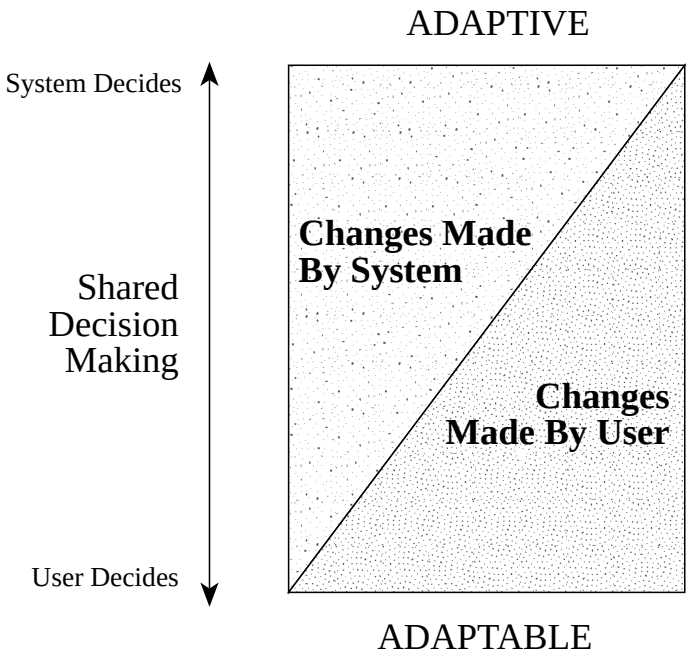


Figure 5.3 Shared Decision Making

5.6.1 Critics

Critic systems have been designed for several domains including LISP code [Mastaglio 1990] and kitchen design [Fischer et al. 1991a; Fischer et al. 1991b]. The general purpose of critic systems is to assist users in performing some type of construction activity. Critics analyze construction situations and emphasize suboptimal aspects of it that users do not see. Users are required to judge the criticism and decide if repairs are warranted (see Figure 5.4). Advising systems [Carroll, McKendree 1987] are similar in nature but do not require users to generate partial solutions. Users describe a problem and advisors propose potential solutions. These systems are primarily designed for one–shot dialogs where advisors answer individual questions.

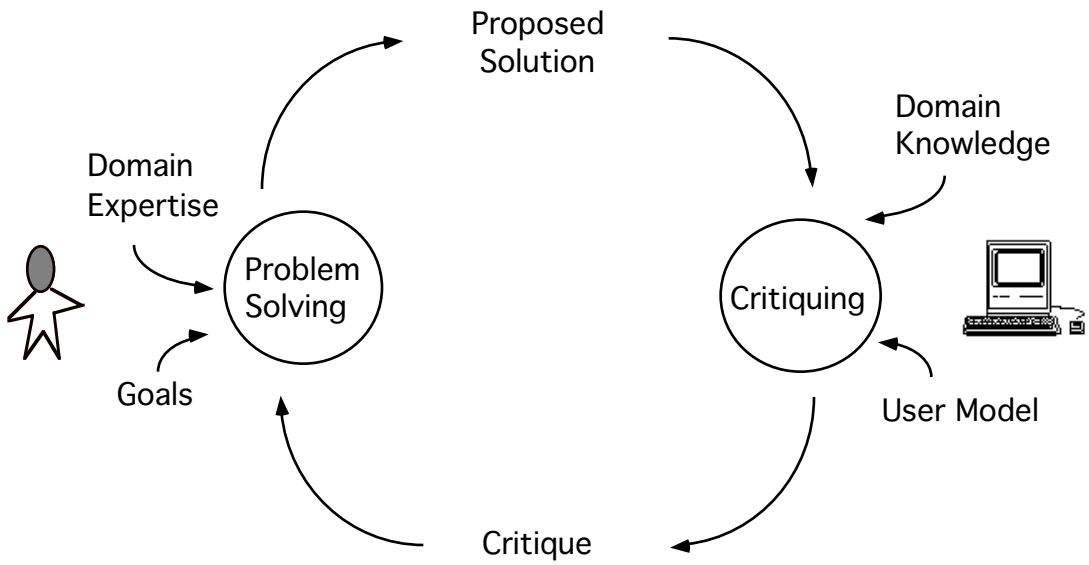


Figure 5.4 Critics

Critic systems are assumed to be in the best position to critique and users are assumed to be in the best position to construct artifacts. The interaction should be structured (unstructured?) so that user and system can switch roles. This allows the system to do construction when appropriate and puts the user in the position to critique what the system develops. This change of roles is the essence of mixed-initiative interaction [Bobrow et al. 1977; Fischer, Stevens 1987].

5.6.2 Agents

INFOSCOPE extends the notion of critics in its implementation of agents. Traditional critics participate in a feedback cycle with the user by offering criticism of artifacts that the user has constructed. INFOSCOPE agents extend this model by allowing user and system to switch roles when appropriate. Agents are able to recognize patterns of interest in keyword terms and patterns of usage of newsgroups. Once these patterns are recognized, INFOSCOPE agents construct filters for the user. This is important because users cannot construct filters for patterns of interest that they are not aware of. Users may then participate in the feedback cycle by offering criticism of the artifact constructed by the system (see Figure 5.5). This is a cooperative process (mixed–initiative) because INFOSCOPE agents are not fully autonomous. They only make suggestions and don't take action until users confirm a desire to have that action take place.

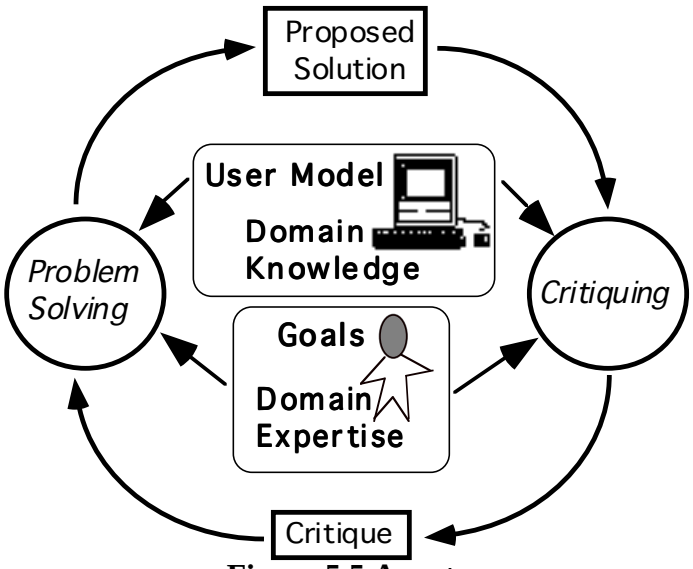


Figure 5.5 Agents

5.6.3 Adaptable Features of InfoScope

Virtual Newsgroups: INFOSCOPE allows users to personally reorganize the information space, adapting it to their own ideas of how information should be organized. Mechanisms required for this include methods for filtering, making the filters persistent, and displaying virtual newsgroups. This is accomplished through the use of CLOS objects, LISP code for storing and executing filters, and a graph display system.

Configurations: One of the problems with standard news readers like RN is that there are few, if any, mechanisms available to organize which newsgroups are displayed and in what order they are displayed in. In the case of RN there is a .newsrc file that contains the order and subscription status of each available newsgroup. However, there are no mechanisms provided to manage this file (for displaying in different order) other than the opportunity to (un)subscribe to individual groups. INFOSCOPE provides users with the ability to define configurations. Each configuration describes a set of newsgroups, and positions for each on the display. For example, I have a Macintosh configuration that shows me only Macintosh newsgroups, a jobs configuration that shows me newsgroups where I might find job offerings, and a software configuration that shows me only virtual groups that concern software products that I currently own. Mechanisms required to provide these services are limited to LISP and CLOS code. In this implementation users can choose a "save configuration" menu item and the system will prompt for configuration a name for the newsgroups currently displayed on the screen. Configurations are saved to a file to make them persistent between sessions.

Preferences: Another adaptable feature of INFOSCOPE are preferences. Preferences are attached to the system and to individual users like the initialization (.newsrc, .login .cshrc etc.) files in UNIX. System preferences control where INFOSCOPE finds Usenet information and mail service on the local network. User preferences control things like the location of a signature file, an email return address, whether or not to

subscribe to any displayed newsgroups and where the newsrc file is located. Using these last two items users maintain compatibility with existing news readers. Preferences attached to newsgroups include sorting options and default storage locations. Preferences are implemented with LISP and CLOS code and made persistent by storing their data to files.

5.6.4 Adaptive Features of InfoScope

Virtual Newsgroups (filters): Filters are adaptive, but not autonomously. Agents post suggestions that are adaptations of the existing information structure. The user must then take some action on those suggestions. This requires a mechanism for estimating the usefulness of existing filters and the potential usefulness of new filters. INFOSCOPE uses rules that are loosely based on Anderson's model of frequency and recency [Anderson 1990]. Anderson's model shows that power functions can be used as excellent predictors of future interest in electronic mail items. The purpose of INFOSCOPE, however, is not to prove that this model works and therefore agent predictions do not have to be as accurate. This is the point of making the filter creation process a cooperative one instead of relying completely on the model's ability to accurately predict information interests. INFOSCOPE utilizes the concepts of frequency and recency but does not actually implement Anderson's model.

Agents: These are also adaptive in a non-autonomous fashion. If users reject suggestions then agents can present a dialog containing information about how & why the suggestion was originally made. Agents know how to adjust the variables used in their calculations, but do not know what the new values should be. Users must help in this regard. For example, an agent might use 5 sessions to determine that a user is interested in a topic. If a user rejects a suggestion based in part on this rule, the user can decide to have agents wait longer before determining interest. This type of user likes to get fewer, higher quality suggestions. Alternately, the user might decide that agents are waiting too long to determine interest in a topic. This type of user doesn't mind critiquing suggestions and want more, potentially lower quality suggestions.

This requires mechanisms for adjusting rule variables and for presenting suggestion justification. Variables used in calculations are stored in working memory objects (which are written to files to make them persistent) and these values are never hard coded into the rules. Hard coding is used, however, in the display of justifications. Rules themselves don't change over time, but each user may adjust the variables the rules use for calculations. Therefore, justification can include canned descriptions of the rules, and the modifiable values in context (displayed as editable text).

6. Related Work

INFOSCOPE is the most recent addition to a long line of message handling systems. This section discusses how several major steps in the evolution of message handling systems have contributed to the development of INFOSCOPE. Several systems that exhibit either strong popularity with users, large numbers of users, or represent significant advancements in technology are discussed. There are four classifications of message systems: basic message systems, graphical direct manipulation systems, knowledge–based systems, and knowledge–based cooperative problem solving systems. In addition, some systems that do not explicitly handle messages, but have contributed to the development if INFOSCOPE in some way are also discussed.

6.1 Messaging Systems

6.1.1 Basic Message Systems

Basic message systems have text–based user interfaces. They supply all of the needed functionality like sending, receiving, deleting, and saving messages. Some of the systems utilize the cursor addressability of intelligent terminals to allow cursor positioning over the text for message selection, but have no other direct manipulation capabilities. Many of the most popular systems fall into this category. Because these system can run on any terminal, they are very popular, but suffer from the same problems as RN (see Chapter 4.3).

6.1.2 Graphical Direct Manipulation Systems

These are message systems including all of the functionality of a basic message systems with the additional use of graphical representations of newsgroups/folders/directories and a direct manipulation interface for carrying out the basic operations. Some of these system (like TRN and GNEWS) present newsgroups in a graphical organization similar to INFOSCOPE. These systems have influenced the development of the INFOSCOPE interface. INFOSCOPE extends these system by graphically presenting messages and compiling them into conversations.

6.1.3 Knowledge–based Systems

These message systems include the ability to define filters which effectively create new structure in the information space. Each of these systems force the users to incur the overhead of filter definition and maintenance (see Chapter 5.4). Examples of such systems are THE INFORMATION LENS and a system developed by M. F. Wyle [Wyle, Frei 1989] that allows users to explicitly save messages to a user model. The system then retrieves items that match the saved items.

6.1.4 Cooperative Problem Solving Systems

These systems, including INFOSCOPE, do not leave the entire task at hand up to the user. They include knowledge about the task which is sufficient for helping users arrive at better solutions to their problems [Fischer 1988; Fischer 1990; Lemke, Fischer 1990]. INFOSCOPE agents actually construct filters based on user actions, requiring only critique of those suggestions from users. At the same time, users may manually define filters as in the knowledge–based systems described above. The difference is that INFOSCOPE agents will monitor these filters to help ensure their continued relevance to the users interests.

Some specific systems deserve special attention with respect to their influence on the development of INFOSCOPE. These systems are discussed in more detail below.

6.2 Interesting Systems

6.2.1 The Information Lens

This system [Malone, Grant, Turbak 1986] is based on an extension of the header field concept. Structured headers, called *templates,* represent different message types and consist of a predefined set of required and optional header fields. Fields are filled by selecting *objects*. For example, an address field in a mail message might be filled with a person object. This object contains several pieces of information about a person, encapsulated within a single representation. In this case the relevant information would be the person's name and electronic mail address. Different templates exist for different types of messages. A meeting announcement, for instance, might need different objects than a memo.

With the INFORMATION LENS, users add structure to messages at send time (see Figure 5.1). This means that the sender of a message selects a template from hierarchy of templates and defines the message type. Then the user access a central library of objects to fill in the template. In addition, there is a significant amount of work to be done before enough objects exist to reduce the constant necessity of creating new ones. While this idea works great in a small environment of users where standards and expectations can be imposed, it may not work well in a global environment where people just want to get work done [Carroll, Rosson 1987]. The reasons for this relate to the cost/benefit ratio as perceived by the sender of messages [Grudin 1989]. Since the sender is not the reader, and since adding structure helps the reader find the message easier, there is no direct benefit to the sender from adding structure at send time. While users must send messages to get their work done and their plans made, a typical sender merely wishes to write the text of a message and send it in as little time as possible.

INFOSCOPE assumes senders will not expend the extra cognitive effort necessary to carefully classify messages or fill out specific templates at send time. It allows users to restructure the information space according to individual semantics at read time. Since this restructuring makes access at read time easier for the reader there is direct benefit perceived and therefore, more incentive to complete the structuring process.

The INFORMATION LENS system spreads its abilities across all steps in the life cycle, but in so doing relies on having users structure messages at send time. This approach ignores the fact that many external information sources lack any sophisticated structure (e.g., AP Newswire) and therefore, systems must be flexible enough to handle unstructured information. This means being capable of presenting a conceptually coherent representation of the information space in question.

6.2.2 Helgon & CodeFinder

The HELGON and CODEFINDER systems [Fischer, Henninger, Redmiles 1991] use retrieval by reformulation and spreading activation techniques to help users retrieve information from an information space without necessarily knowing exactly what it is they are looking for. However, in order for retrieval by reformulation to proceed, the information space must first be highly structured. In the case of HELGON, this structure takes the form of a subsumption hierarchy. Encoded within this hierarchy are the semantic relationships between objects and their individual classifications. Again, we must ask where this structure comes from. Here HELGON is much like a traditional database since a database administrator creates its underlying structure (see Chapter 5.3).

The HELGON system has capabilities in the storage time and question time domains (see Figure 5.1). At question time it allows users structured access to an information space. If the structure presented is misunderstood by the user, or if the user is not sure of the precise goal, a reformulation process can be carried out to map the query into the system model. The CODEFINDER system adds the ability to spread activation values across a network of nodes to retrieve items *related* to a selected item. At storage time these systems allow the user to store modified versions of recovered artifacts, possibly changing the semantics of the artifact in the process.

HELGON contributes the method of retrieval by reformulation to INFOSCOPE. An initial definition of a filter in INFOSCOPE can be just as incomplete as an initial query in HELGON. This occurs because of two facts: (1) just as in HELGON, users defining a filter may not know exactly what it is they are interested in, and (2) users' interests may change dramatically over time. Both of these situations require reformulation of the query. In HELGON, the system retrieves examples from the information space that match the partially formulated queries. These examples are used to guide the user towards the desired query. In INFOSCOPE, the messages retrieved by a filter may initially include items that are not really of interest. When those messages are ignored or deleted the need for filter reformulation arises.

6.2.3 InVision

The INVISION system [Kass, Stadnyk 1992] addresses the problems of communications in large organizations. It is an intelligent distribution system for mailing lists. Instead of having to maintain a set of distribution lists the system uses information about the potential recipients to dynamically compute the appropriate recipients of messages; in this case engineering release announcements. The system is able toanswer two key questions for users. The first of these is the "Who do I tell? question. Given a message, the system consults a knowledge–base of user interests

and constructs a list of potentially interested readers of that message. This works well because INVISION operates within the confines of a single organization in which the structure and content of messages is well defined and highly structured. In Usenet, one never knows who message recipients are.

Because the system has information about recipients, it can also answer questions concerning "who do I ask" about a certain thing. In Usenet News, users broadcast to a topic category and assume that only the appropriate people would be reading that group. "Who do I ask" and questions are not as crucial in Usenet (due to vocab problem etc.) as is the "Where do I find it" question. Finding information in a huge global bulletin board is different than any message distribution system that is limited to either a single organization (due to the shared knowledge base of terms and interests) or a single type of structured information (due to the additional leverage this gives you for sturdier automatic classification techniques).

7. A Scenario

This section presents a scenario of INFOSCOPE usage to help make the interaction style of INFOSCOPE, its relationship to the conceptual framework, the evolution of the user model, and the problems that INFOSCOPE tries to solve more explicit. Many of the situations described are taken from the user studies discussed later (see Chapter 9).

7.1 Setting Preferences

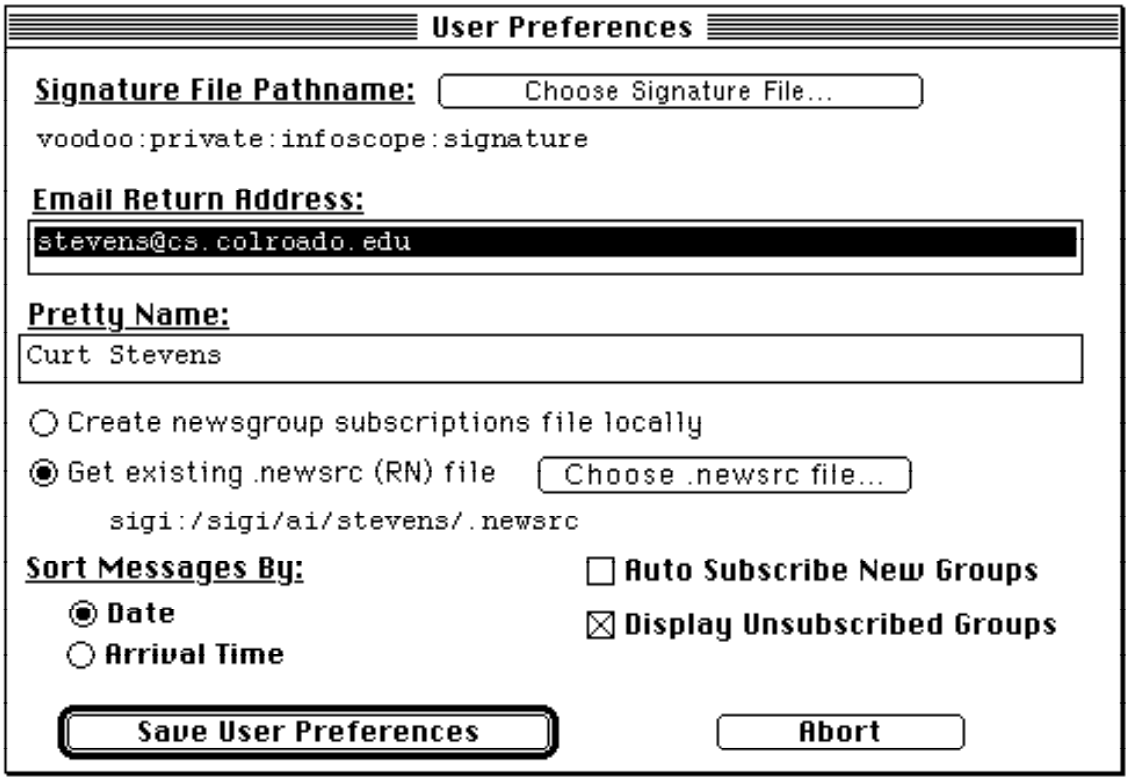


Figure 7.1 User Preferences

This dialog solicits information from the user that is necessary to get INFOSCOPE up and running for first time users. The values can be modified at any time by selecting a menu item.

The first time users launch INFOSCOPE they are required to fill out preference dialogs that cover several areas. Along with some preferences that have little to do with the conceptual framework of INFOSCOPE (other than the fact that they are necessary to properly run the system), users must specify whether or not they will be using an existing .newsrc file (see Figure 7.1). This is important because it will determine what newsgroups are contained in the users' initial default configuration (see Chapter 5.4.2). An example of a default configuration is shown in Figure 7.5.

However, empirical evaluations demonstrated that users commonly subscribe to many newsgroups that they never read and these subscriptions are transferred from the newsrc file to the initial configuration. This is a limitation of using newsrc files to seed INFOSCOPE's knowledge of the user. When no newsrc file is supplied users are presented with the top level of the Usenet hierarchy as their default configuration (see Figure 7.2).

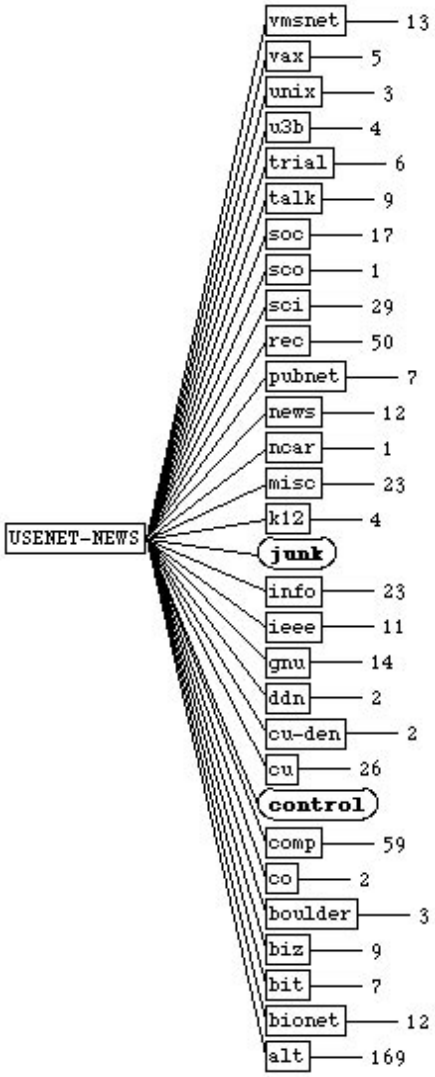


Figure 7.2 The Usenet Top Level

When users do not supply a .newsrc file to seed INFOSCOPE's knowledge of subscribed newsgroups, this default view of the Usenet top level is used. Square nodes represent partial classifications that contain other classifications or newsgroups. Oval nodes are actual Usenet newsgroups that contain messages.

INFOSCOPE also looks for a KILL file where the newsrc file is kept. KILL files contain complete subject lines from conversations that users have previously asked RN to eliminate from all message displays. This information is used to fill out the exclude portion of future filters. The idea is that if the user has already specified certain terms that should be excluded from newsgroups, INFOSCOPE should use this

information when suggesting filters. While this works in theory, none of the participants in the INFOSCOPE evaluations had existing KILL files for the system to utilize.

Another preference that users are required to fill out is one that controls the frequency of suggestions to create new configurations (see Figure 7.3). This directly controls the intrusiveness of agents that make these suggestions.. Users have control over two parameters for this purpose. The first controls the percentage of recent sessions in which a newsgroup must be browsed to be included in a suggestion. The larger this number, the less intrusive the agent. The second parameter sets the number of sessions that are considered recent by an individual user. For a given percentage of sessions, the larger this number the less intrusive the agent. In this scenario, the user has declared that configurations are to be suggested whenever newsgroups are browsed in at least 30% of the last 10 sessions.

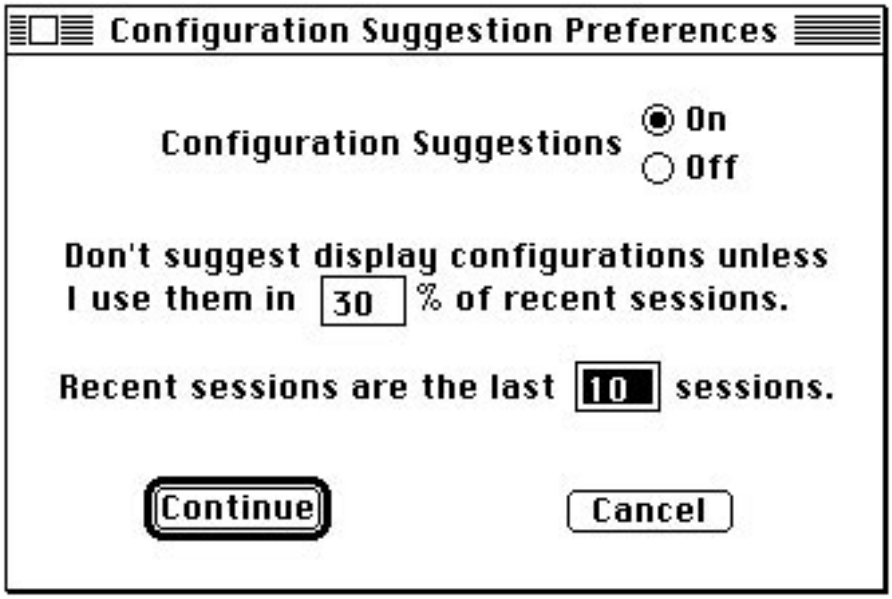


Figure 7.3 Configuration Preferences

Users fill out this dialog to determine the parameters agents use to make configuration suggestions.

As mentioned earlier, the concept of noise in Usenet news is related to the number of uninteresting messages individual users must browse while searching for interesting messages. However, the number of uninteresting messages that constitute noise for any individual user may vary [Klapp 1986]. For this reason INFOSCOPE provides a mechanism for users to adjust the measure of noise. The final preference dialog users fill out controls the parameters of agents that suggest virtual newsgroups (see Figure 7.4). The first two parameters are similar to the ones that control configuration preferences. One controls the frequency of newsgroup browsing in recent sessions that triggers a suggestion. The other controls the users' definition of recency by setting the number of sessions agents will look at when calculating frequency. The third parameter sets how often a term (keyword) must appear in read messages before it is included as a term in a suggestion. Using these parameters users control how intrusive agents will be in suggesting virtual newsgroups. When suggestions are

rejected, users may adjust these parameters and have the agents reconsider the suggestion when the new parameters are met. Some users take the time to make these adjustments while others simply set them up and ignore them when rejecting suggestions. The final parameter for virtual newsgroup suggestions eliminates terms from suggestions if they come from only one conversation. This prevents terms of passing interest from large conversations from being included in future suggestions.

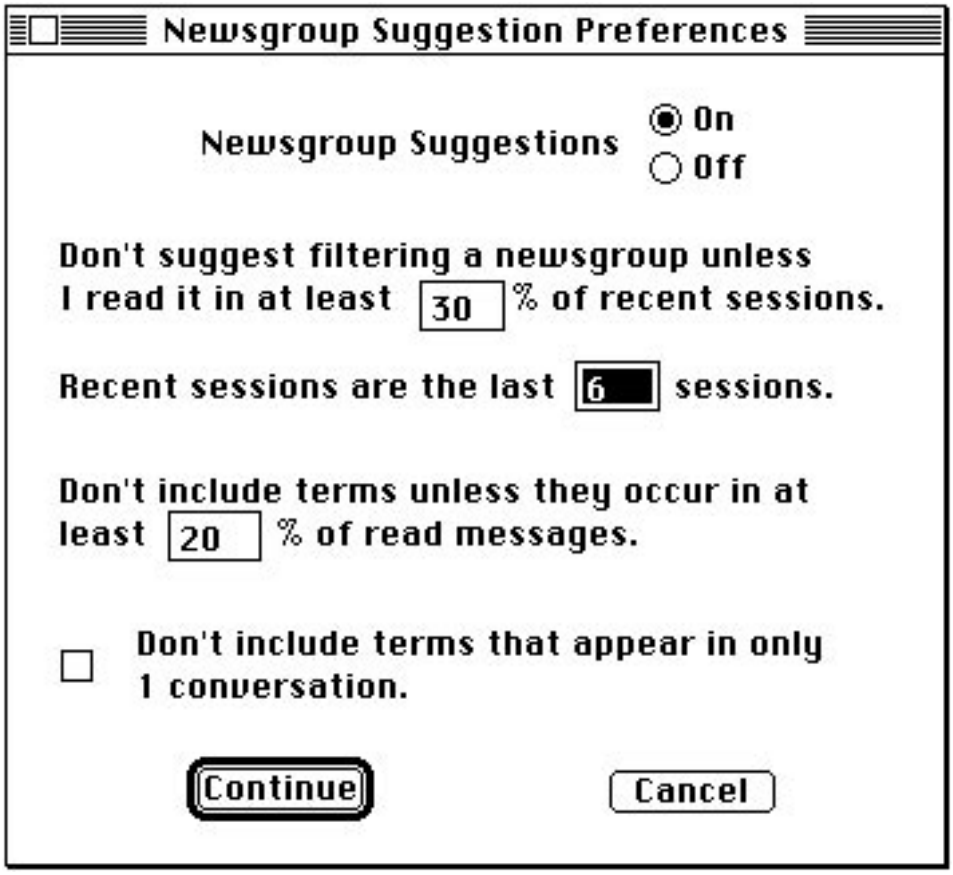


Figure 7.4 Newsgroup Suggestion Preferences
Users fill out this dialog to determine the parameters agents use to suggest virtual newsgroups.

7.2 Browsing Newsgroups

A main activity for users in any Usenet news reader is browsing newsgroups. INFOSCOPE presents users with a graphical representation of the available newsgroups. When users choose a newsgroup to browse they click the mouse on that node and select the action “browse messages” from the resulting menu. Doing so results in the display of the messages within that newsgroup. One problem with the use of browsers in this manner is that people have trouble browsing large information spaces, especially if that space is poorly structured [Halasz 1987]. While there is no way to completely eliminate these problems in an information space the size of Usenet News, INFOSCOPE attempts to reduce the negative impact of them in two specific ways. First, the size of the information space is effectively reduced by the definition of configurations (see Figure 7.5).

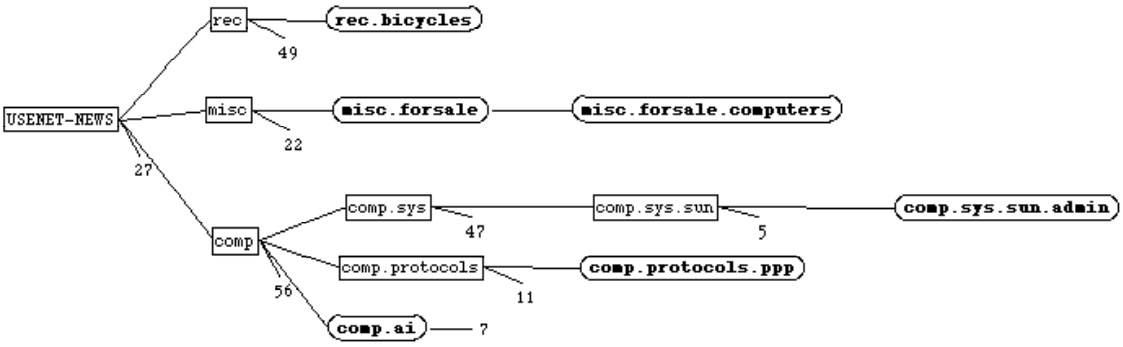


Figure 7.5 A Default Configuration

This is the default configuration for one of the subjects in the INFOSCOPE evaluations.

The initial configuration for the user represented in this example contained the 13 newsgroups that were marked as subscribed in his RN newsrc file. However, during the first two weeks the user only browsed messages from a subset of those newsgroups. The user model kept track of the newsgroups actually read (see Table 7.1) and the user was presented with a suggestion to create a configuration containing only those groups.

Table 7.1 Configuration User Model

User	Subject–A
Displayed Newsgroups	misc.forsale.computers comp.dcom.sys.cisco rec.bicycles comp.sys.sun comp.protocols.ppp comp.protocols.tcp-ip comp.sys.sun.admin comp.sys.sun.hardware comp.sys.sun.wanted comp.dcom.lans.misc comp.dcom.lans.fddi comp.dcom.lans.ethernet comp.dcom.lans
Browsed Newsgroups	rec.bicycles (1, 2, 3, 6, 7, 9, 10) misc.forsale.computers (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) comp.sys.sun.admin (3, 5, 6, 8, 10) comp.protocols.ppp (8, 9, 10) comp.ai (1, 5, 9, 10)

The portion of the user model concerned with tracking configuration suggestions is represented above. The numbers in parentheses indicate the session numbers in which that newsgroup was browsed. This example led agents to suggest a configuration displaying only those newsgroups that were browsed.

The second way that INFOSCOPE helps reduce the deleterious effects of so much information is to provide mechanisms for efficiently browsing messages.

7.3 Reading Messages

Aside from browsing and manipulating the structure of newsgroups in INFOSCOPE, users can display and navigate through the set of messages contained within any newsgroup. When users click on newsgroup nodes they are presented with a window containing those messages (see Figure 7.6). In this example, some of the messages for the newsgroup misc.forsale.computers are shown. Several newsgroups can be displayed at the same time allowing users to browse whatever newsgroups they want in any order.

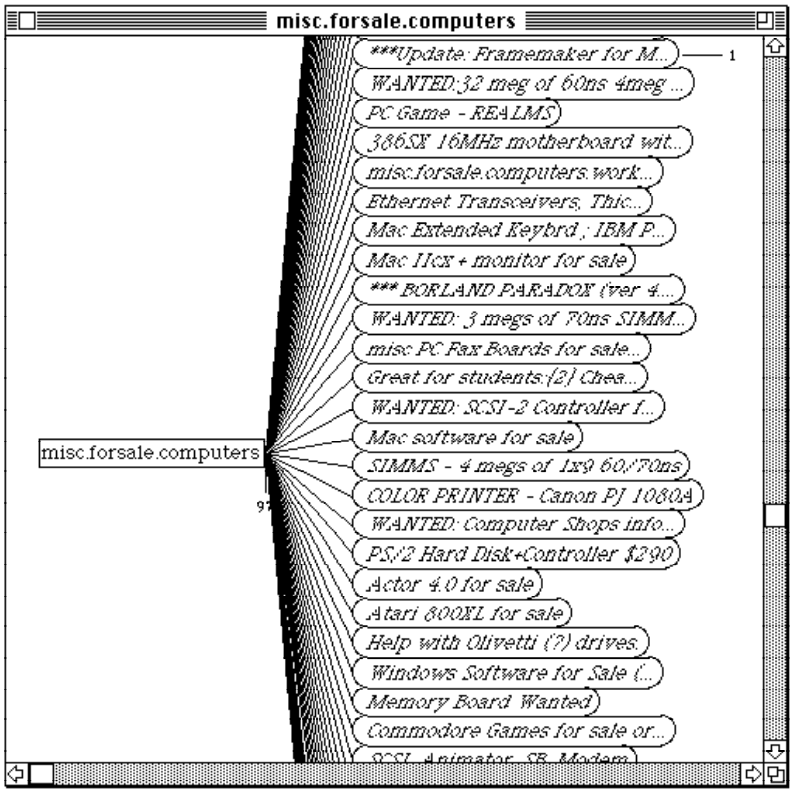


Figure 7.6 Browsing Messages

Messages are displayed in a hierarchical organization, similar to that of newsgroups, to aid users in understanding the structure of posted messages. In this way messages are organized around the concept of a *conversation.* A conversation consists of a message node, all responses to the message in that node, and similarly all responses to the responses until leaf nodes are reached. This idea is demonstrated in Figure 7.7 by the conversation with "********** 386DX/20 M/B …" displayed in its node. This is the subject of the conversation. Notice the nodes that start with "Re:" and are attached to a parent node. These messages are responses to the original posting. "Re:" stands for 'regarding' and is included in the subject line of a message by standard news software whenever a reply is posted. Messages in response to a response will have "Re: Re:" at the beginning of the subject line and the pattern expands as long as the conversation continues.

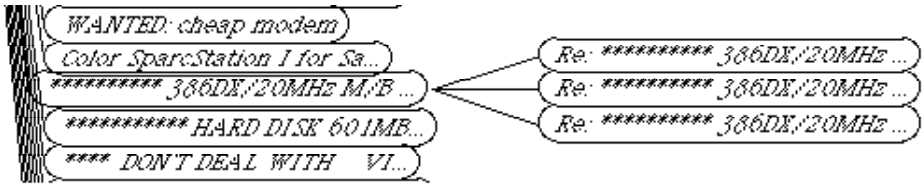


Figure 7.7 A Conversation

There are several advantages to displaying the structure of messages in the conversational metaphor. In the INFOSCOPE system conversations may be treated as an object. This means that operations such as "Save this conversation," or "Monitor this conversation" can write all the messages of a conversation to a file at once. If a conversation is monitored then the system automatically saves new messages in the conversation to the same file as the original conversation. By monitoring a conversation, users can leave on vacation and not miss any messages from that conversation. Another advantage is that this makes saving subconversations as simple as saving entire conversations. For example, if someone responds to a posting with an interesting comment it may start a whole new conversation within a conversation (a new thread). This thread can be saved as a conversation by clicking the mouse on the node at the root of the thread (i.e., the message with the interesting comment). Also, since some interesting threads within a conversation will lead to deep message hierarchies starting from certain nodes (i.e., many responses), the display of this structure can immediately lead users to these interesting threads. Likewise, uninteresting comments, and the useless replies identifying them as such, can easily be pruned from the conversation in the same manner. For example, users often wish to prune the many FLAMES created in response to a stupid comment.

Messages (whether being read or written) are displayed in completely editable text windows (see Figure 7.8). This allows terms to be freely copied from messages and pasted into filter definitions. In addition, as users read, save and respond to messages the user model keeps track of the headers of those messages. In INFOSCOPE users have several methods of indicating to the user model that certain messages are of particular interest. Interest information is implicitly added to the user model whenever messages are deleted from the display (indicating negative interest), saved to a file or replied to (indicating positive interest). When users save messages to a file, the terms used in the file name are included in the user model. In addition, users may explicitly indicate interest by holding down modifier keys when skipping to the next message or by selecting individual words and selecting a menu item. All of this is indicative of the general agent approach in INFOSCOPE. Users can participate in the development of the user model to whatever degree they are comfortable with.

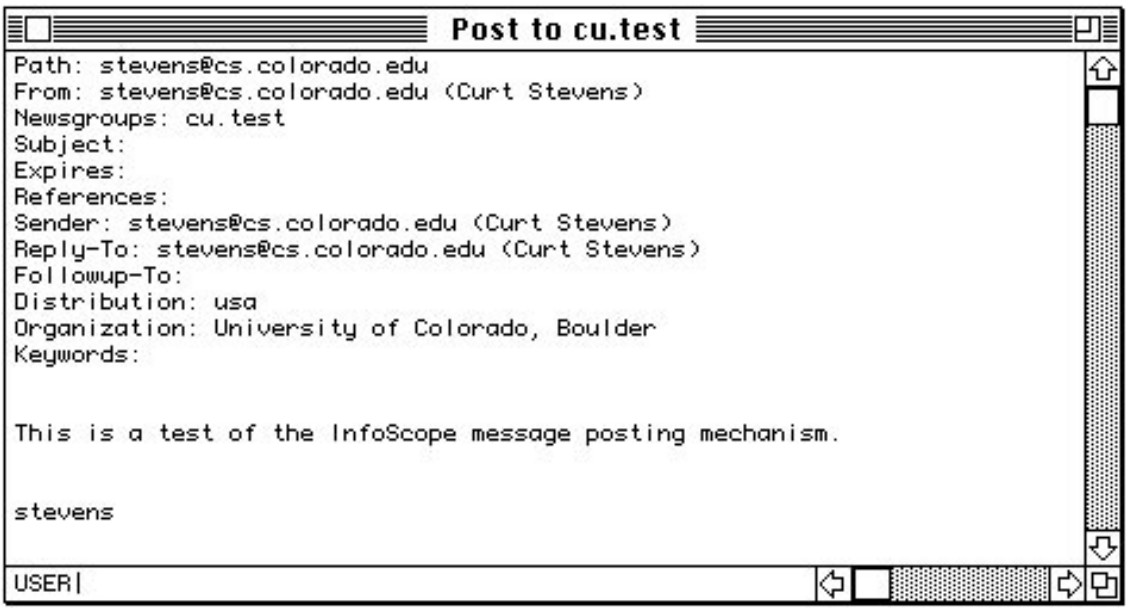


Figure 7.8 Posting Messages

Initially, INFOSCOPE writes the header information, any user initiated indicator of interest, and the terms from filenames of saved messages to a file. This information then passes through a filter (a stop list) in order to exclude terms and header fields that contain no semantic information. The resulting terms are then passed onto the user model. A partial snapshot of a user model is shown below (see Table 7.2).

Table 7.2 Term Interest User Model

User	Subject–A
Newsgroups	misc.forsale.computers
Terms	subject: imagewriter (+5), macintosh (+12), sale (+23), Mac (+31), zoom (+2), wren (+1), v.32bis (+2), seII (+2), quadra (+4), forsale (+17), telebit (+3), IIcx (+7), mac2 (+12), lc2 (+5) from: Xxxx Xxxxxxxxx (-1)

A portion of a user model concerned with tracking newsgroup suggestions is represented above. The numbers in parentheses indicate the current level of interest in that term. This example led agents to suggest a newsgroup containing some of the terms shown above.

7.4 Defining Filters and Handling Suggestions

Creating filters manually and modifying suggestions that create filters are essentially the same process. The major difference is that user and computer switch roles. In manual filter creation users play the role of designer, allowing the user model to make suggestions that keep these filters relevant to current interests. When creating suggestions agents play the role of designer and users play the role of filter critic. This is consistent with the underlying concept that users should be able to serve in the role of designer or critic whenever appropriate. The system is both adaptable and adaptive. As the system model changes to more accurately reflect the needs of individual users, each user can view their information space through their own situation models.

To initiate the process of creating virtual structure users click the mouse on any newsgroup that will be one of the parents of the virtual newsgroup. Parents can be either real or virtual newsgroups. When users select either “Create Virtual Newsgroup” or “Edit Virtual Newsgroup” they are presented with a partially filled out filter. INFOSCOPE uses whatever information it has to complete the filter. When creating a new newsgroup, it fills in a default name for the new newsgroup and the parent that was used to enter the restructuring process. In this scenario, if the user were to initiate filter creation by clicking on the misc.forsale.computers newsgroup, INFOSCOPE would present a filter with the name misc.forsale.computers.virtual and misc.forsale.computers would be filled in as the parent of the new virtual newsgroup.

Alternately, if the user never creates a virtual newsgroup inheriting messages from misc.forsale.computers, the user model described in Table 7.2 triggers an agent that suggests one. Using the information contained in the user model agents suggest the filter shown on the left side of Figure 7.9. This is an actual suggestion made to a subject in the evaluation studies. Referring to the user model you can see that the components of the suggestion came directly from observations of user reading patterns. Several of the items in the model were not included in the suggestion because they did not satisfy the parameters set by the user for the suggestion of virtual newsgroups. The user proceeded to modify the suggestion by removing several terms and adding to new ones.

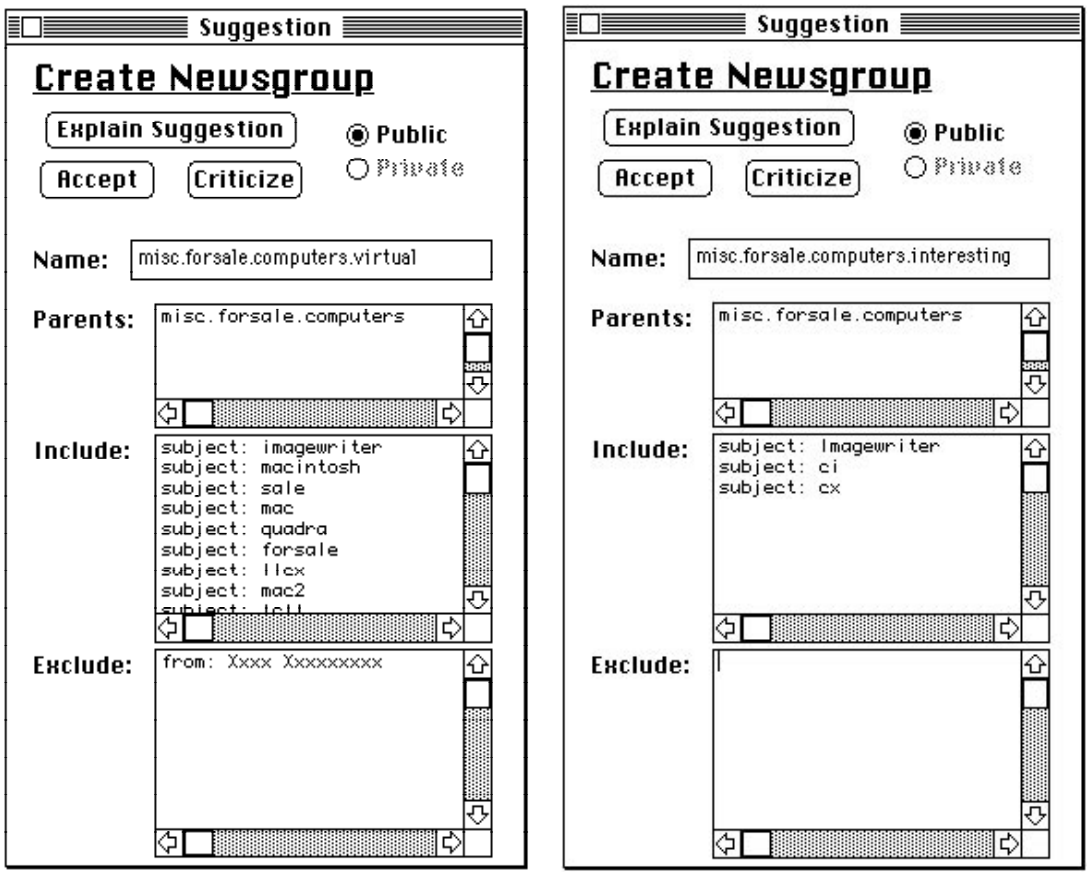


Figure 7.9 A Sample Suggestion

This is what a suggestion looks like to the user. The suggestion on the left is the one supplied by agents and the suggestion on the right is the same suggestion after being modified by the user. This degree of change was typical in the INFOSCOPE evaluations.

The right side of Figure 7.9 shows the final filter after user modification. When the filter is accepted, the active configuration reflects the existence of the new virtual newsgroup (see Figure 7.10) In addition, this user is now able to browse a newsgroup that contains a significantly higher percentage of messages relevant to his search for specific computer equipment (see Figure 7.11).

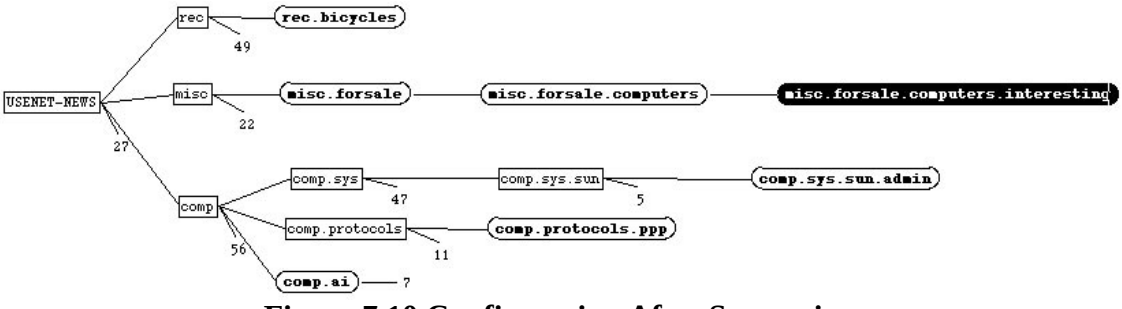


Figure 7.10 Configuration After Suggestion

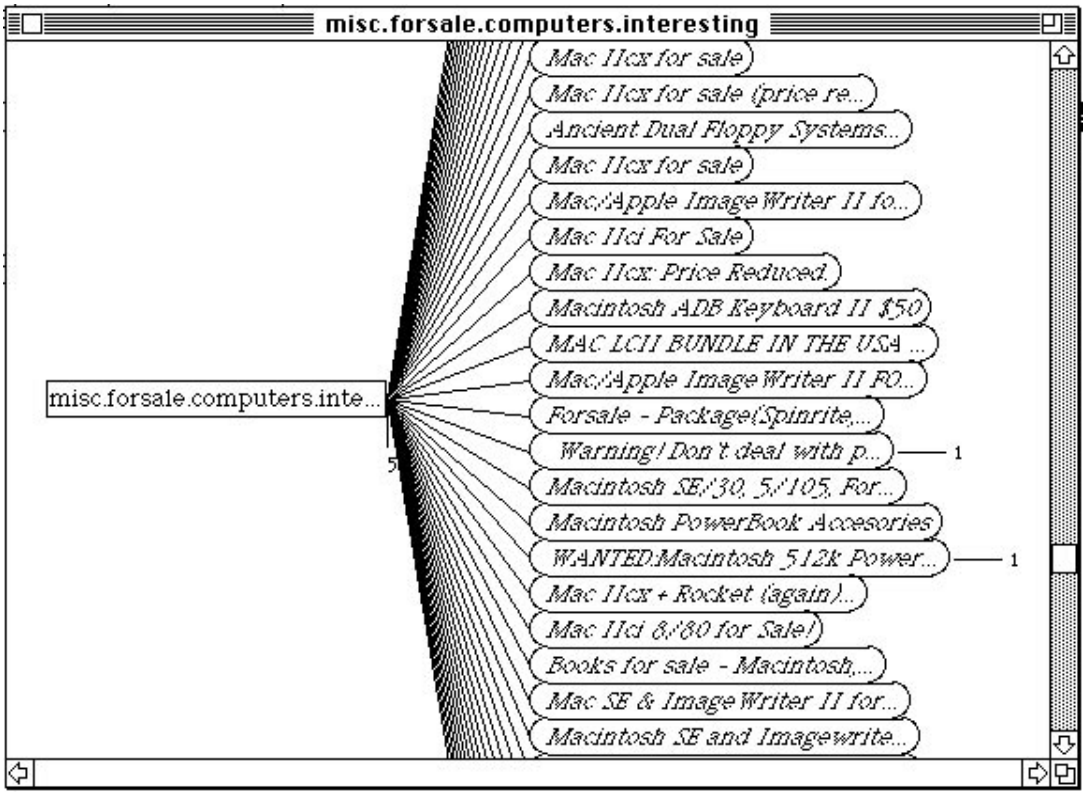


Figure 7.11 A Virtual Newsgroup

8. Design Methodology

8.1 Personalizing the Information Space

In order to investigate solutions to the problems discussed so far (see Chapter 5), the INFOSCOPE system provides two mechanisms for personalizing the information space. Users may define configurations that are personal views into the information. Since there are almost two thousand newsgroups, this allows users to concentrate on the information sources that are most likely to contain interesting information. However, even inside newsgroups of potential interest there are often far too many messages to browse. To approach this problem INFOSCOPE provides tools for defining virtual newsgroups. A virtual newsgroup refines the a priori Usenet hierarchy by filtering out messages regarding specified topics. Filter definitions can include or exclude messages based on the contents of any header field. This reduces information overload by creating smaller newsgroups containing relatively narrow ranges of coverage. Virtual newsgroups reduce the impact of the vocabulary problem by allowing users to define personalized mappings from keywords to group names. In addition, virtual newsgroups are not limited to a strict hierarchy. By defining filters that search several parent newsgroups, users create a directed graph. A priori newsgroups with similar topics may be combined, and even filtered again to create a totally personal organization. Also, by analyzing the structure added by each user, some of the structure needed by retrieval systems like HELGON (see Chapter 6.2.2) can be added automatically. That structure will not suffer from the same problems as a priori structure since, by definition, it consists of terms known to the user. Finally, analyzing interactions with the information space indicates shifting interest patterns that are used to help maintain the virtual structure.

Personalization of information spaces is problematical for two main reasons that are addressed by this research. First, it takes a non–trivial amount of effort to engage in personalization and second, users only carry out personalization for patterns of interest that they explicitly recognize.

8.2 Reducing Personalization Effort

Soon after virtual newsgroups were implemented it became clear, through my own use of this mechanism, that managing the necessary filter definitions could be just as much, if not more effort than manually searching for interesting messages. Research has shown that the maintenance and evolution of information filters is a long term task [Mackay 1990b]. In fact, there has been a steady pattern of attempts to simplify news reading that each led me to successive strategies requiring user effort (see Table 8.1). Newsgroups were originally an answer to the chaotic mess created by so many messages. This led to an a priori structure that is difficult for readers to use. To solve this, virtual newsgroups were implemented and that led to a need for managing filters.

Table 8.1 Effort Reduction Strategies

Cause of Effort	Effort Reducing Technology
Usenet News	subject classification (newsgroups)
a priori structure	personalized structure (virtual/filtered newsgroups)
filter management	agents, *suggestions*
agent management	situated feedback
suggestion management	*effort takes place in situation model*

While researching the problems of information access in the Usenet News environment, it became obvious that each attempt to reduce the effort caused problems which actually increased the effort. The INFOSCOPE system is based on the idea that requiring user actions in their situation model rather than the system model effectively makes the system easier to use and comprehend. With the introduction of agents, users manage *suggestions*, that are in the situation model, rather than managing agents, filters, or a priori structure, all of which are in the system model.

INFOSCOPE addresses the filter management problem by implementing *agents*. These agents monitor user interactions and help with the tasks of creating and maintaining virtual newsgroup filters. In order to make maintenance easier, agents post suggestions that are based in the user's situation model. This is possible since the system analyzes past interactions and behavior patterns to determine what virtual structures have been used to fill the situation ↔ system model gap. Since the suggestions are in a situation model (i.e., expressed in the terms that users actually read about), users can understand them. By transferring the necessary work into the situation model, users spend less time mapping between models and more time reading interesting messages. Also, some users prefer intrusive agents that make many suggestions and others prefer benign agents. Users can criticize suggestions and modify the criteria on which suggestions are based. So, agents transfer the users' task to that of perusing the information space and managing suggestions. A more detailed discussion of agents can be found in section 8.6.

8.3 An Overall System Architecture

The INFOSCOPE system is implemented on the Macintosh platform. This environment provides a standard COMMON LISP [Steele 1990], an implementation of CLOS (Common Lisp Object System) [Keene 1989], and extensions for supporting the Macintosh interface. In addition, INFOSCOPE utilizes an interface extension to COMMON LISP called CLIM (Common Lisp Interface Manager). In CLIM mouse sensitive objects (presentations) are automatically highlighted and a complicated arrangement of commands defines several contexts in which different presentations are sensitive. Nearly all objects in INFOSCOPE have an internal representation and an external (screen) representation. Internal representations of agents are implemented as OPS5 production rules and the patterns derived from session data are OPS5 working memory elements. The only external representation of agents is the suggestions they

make. Newsgroups and messages are represented internally as CLOS objects and externally as CLIM presentations. Session data have no external representations (beyond the filters they help create) and are internally represented by CLOS objects. Other external objects are represented by standard Macintosh interface objects (i.e., dialog boxes, menu items etc.). Persistent data are written to disk as COMMON LISP code or CLOS objects.

These sub–systems are used to implement the three individual components[4] of the overall INFOSCOPE architecture (see Figure 8.1):

8.3.1 The InfoScope User

The user of the INFOSCOPE system is the center of activity. INFOSCOPE keeps track of many aspects of day to day use of the system. For example, as users interact with newsgroups, messages or filters the underlying CLOS objects record the details of these interactions. Similarly, information about manual filter definitions is gathered. This data is stored in the user model for analysis by agents This analysis occurs off line to improve the response time of INFOSCOPE during interactive sessions. Users fill the role of critic to agents. Suggestions are presented to the user by making them available on a *Suggestions* menu. When no suggestions are available, the menu item is dimmed. Once suggestions are made, the menu is activated and users select them from this menu (see Chapter 7). User criticism is provided in two ways. When users disagree with aspects of a suggestion, they edit that suggestion to conform to their perception of information needs. Also, users can adjust the parameters that determine how confident agents must be before making future suggestions.

4 A fourth component is the NNTP News server. Since INFOSCOPE does not implement a server, but simply utilizes the NNTP protocol as a client, the News server is not included in this discussion.

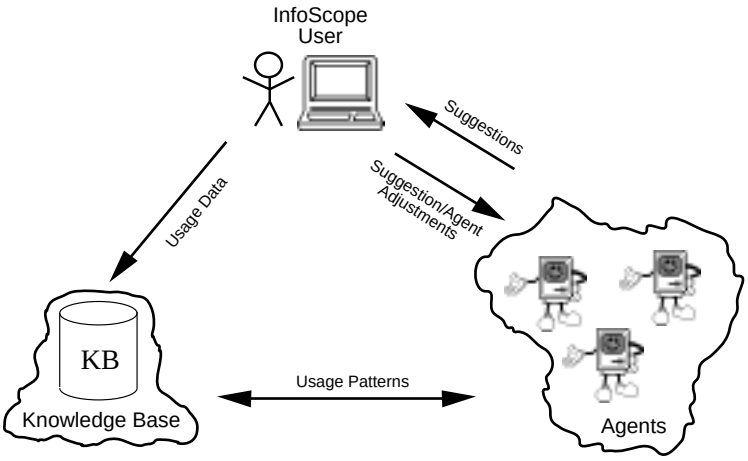


Figure 8.1 System Architecture

The INFOSCOPE system contains four components (user, INFOSCOPE interface; knowledge–base, and agents). The user reads messages and data about these interactions is stored in a knowledge–base. Agents recognize usage patterns and make suggestions based on them. When suggestions are rejected or modified, users can adjust the offending agent's parameters.

8.3.2 The InfoScope Knowledge–Base

As users interact with INFOSCOPE, representations of those interactions are stored in the knowledge–base. This data is made available to agents for the purpose of constructing suggestions. Examples of knowledge stored include the fact that 10 out of 25 messages were read in newsgroup X on day Y, or specific indications of the topics that were read. An issue of crucial importance to the viability of this approach is the effects of age on the usefulness of this data. If all interaction data were to be stored indefinitely, the amount of data quickly overwhelms the ability of agents to process it. Instead, INFOSCOPE agents must attach values to information that decay over time. When the value (or usefulness) of information decays below a threshold value, that information is discarded from the knowledge–base. This does not preclude this knowledge from becoming highly valued again in the future.

8.3.3 InfoScope Agents

INFOSCOPE agents are rule–based heuristics designed to help users manage the INFOSCOPE environment. They must fill two roles in their operations. First they must regularly scan the knowledge–base and recognize interaction patterns that indicate, for example, sub–optimal newsgroup organization. These patterns are used to populate the user model. In their second role, agents must utilize the information contained in the user model. Data within the patterns is used to construct suggestions. These suggestions are subsequently presented to the user in the form of completed filters. A more detailed scenario of agent operations was given in Chapter 7.

8.4 Information Structures Interpreted by InfoScope

Specific sources of knowledge to INFOSCOPE include the newsrc file (keeps track of previously seen messages), information about existing newsgroups that is available

through NNTP [Kantor, Lapsley 1986] (Network News Transfer Protocol: contains lists of active newsgroups, new newsgroups, available messages and their id numbers), the filters that are defined over time, the information contained in the header fields of interesting messages, the user model that contains statistics and interest profiles for individual users and newsgroups, as well as the heuristics comprising the agents themselves.

The most basic, and well understood, source of information is the newsrc file currently in use by a user the first time they start INFOSCOPE. This is the file used by RN. It keeps track of which messages are currently available in the information space (e.g., which messages have not yet expired) and which of those available messages have already been read. Another piece of information kept in this file is data about which groups are subscribed to by individual users. Initially, the system uses this information as an indication of the existing newsgroups that contain interesting information for the user. INFOSCOPE does not change the format of this file so, users may continue to read news from their normal home machines when that is convenient. INFOSCOPE reads newsrc data across a local area network; so that the same newsrc file is used by INFOSCOPE and programs like RN.

This decision to keep new data files compatible with old ones means that additional knowledge structures had to be built. The most important of these is the list of virtual newsgroups defined by users and agents. This structure contains the names of each virtual newsgroup, the filter that defines which messages are inherited into those newsgroups, and the set of locations (real and virtual) from which they originate. This in turn is used by the display system to build a visual representation of the information space structure on the screen.

Another source of useful information to the system is the displayed hierarchy itself. When users display a newsgroup on the screen they may be indicating an interest in messages from that newsgroup. Prolonged use of that newsgroup clearly indicates this. Therefore, the user's actions with respect to organizing the display are gathered by interested agents and stored internally. The virtual hierarchy itself is another valuable source of knowledge about the user. The vocabulary used to name these structures may be associated by agents with any messages filtered into that newsgroup. The combination of the hierarchy and the virtual hierarchy constitute the main portion of the system model that the user must deal with. This source of information, however, produced some interesting research question as well. Most importantly, *what can really be said about a newsgroup or message that is displayed, but not dealt with in other ways?* For example, if a newsgroup is displayed and some of its messages are displayed what can be said about degrees of interest in that newsgroup? If none of the messages is actually saved to a file, INFOSCOPE may not be able to determine anything. This stems from the fact that a user may display a message simply because the header information was not enough to determine the interest level. After seeing the text of the message a user may decide that it wasn't about what they thought at all. Only when a message is explicitly saved or deleted, or when several messages of a conversation are examined, may INFOSCOPE agents be totally sure that a degree of interest has been assigned by the user. This problem also exists in use of the newsrc file for initial priming of the user model. Empirical studies carried out as part of this research indicate that most users subscribe to numerous newsgroups from which they never read any messages. A perceived future interest in

the information contained in those newsgroups never materializes due to either misconceptions about what is contained in them or lack of time to engage in browsing them.

Lastly, there is the knowledge that is garnered by INFOSCOPE from the message storage process. Each time a message is stored it must be placed somewhere specifically. This location has a path and a name that will utilize terms from individual situation models. These terms should therefore be used whenever appropriate in agent suggestions etc.

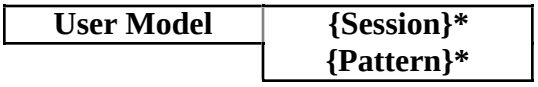
8.5 User Modeling in InfoScope

The user models in INFOSCOPE are concerned with two main aspects of system usage. The first of these deals with the user's individual preferences about the daily operation of the system. This includes which baskets should be initially displayed by default, whether entire subhierarchies of conversations should be displayed by default or only on demand, where the system should read the newsrc from, how often the news stream should be reset and newsgroups updated to their new state and many other aspects of day to day operation.

The second aspect of system usage monitored by the user model is the actual content of information that is deemed *interesting* or *uninteresting*. Interest is implicitly indicated by saving or replying to messages, and explicitly indicated by specific keystrokes when browsing from one message to the next. The monitoring task, as designed in INFOSCOPE, is a statistical one that requires the model to keep track over long periods of time of what users are reading and what they are ignoring. This happens on two levels. At the higher level the system monitors the amount of interesting information that is found regularly within specific newsgroups. When the amount is too low, suggestions are made that modify the existing structure by creating filters for the interesting information. At the lower level the topics covered by messages are retained to facilitate the filter evolution process in a semi-automatic fashion. From these two parts of the user model, INFOSCOPE acquires relevant information about individuals who use it regularly.

8.5.1 The User Model Object

This section describes the user model at the implementation level. The user model is a hierarchically arranged set of CLOS objects that contain information necessary for the effective operation of agents.



A *user model* is a combination of several data structures that are populated with information as INFOSCOPE is used. A user model has two top level components. These are a list of *sessions* and a list of *patterns*. Sessions are defined as everything that happens in INFOSCOPE between launching the system and shutting it down. Some users prefer to keep INFOSCOPE

running all day long and each day is therefore one large session. Other users launch and quit the application several times in a single day and produce many more sessions. Session data is compiled exclusively by the INFOSCOPE system itself and is not effected by agents.

Conversely, patterns in the user model are created exclusively by INFOSCOPE agents. Agents run outside the time frame of sessions (over night), compile their data into patterns and then populate the user model with these patterns and insert suggestions into the INFOSCOPE system. Suggestions are available to users the next time they launch the system.

8.5.2 Sessions

Session	date / time session duration configuration uses configuration changes **{Useful Message Data}***

Sessions keep track of interesting data about each interaction session a user has with INFOSCOPE. The date/time of the session is used to order the information so that time based patterns of information interest can be calculated by agents. The duration of sessions is important so that the level of usage by each user is available. Without duration information, the number of sessions of each user is less meaningful. The set of configurations that are used, changed, added or deleted is important in the user model because that is the basis on which agents can suggest modifications to these configurations. Every user modifies at least one configuration. This is because the default configuration supplied to users in their first session does not contain any newsgroups. Only higher level classification nodes are contained in the initial configuration because of the limited usefulness of newsrc information described earlier. Any interesting message headers are contained in the session data as a sub component of the session object.

Useful Message Data	**Interest Indicator** message header storage file pathname

Included in each session object is an object that contains information about the messages that are considered useful for agent analysis. There is one of these objects (contained in a list) for each interesting message. One component of this object is an indicator of why an individual message is interesting. The extensible set of indicators includes *deleted*, *read, saved*, *saved with conversation, explicit* and *ignored*. Explicit

interest is expressed by users by holding a set of modifier keys when going from one message to the next. Ignored messages may be interesting because a message that matches a filter but is ignored may indicate that the filter needs refinement. The message header itself is important because this is where terms for filters are obtained. The last piece of information included is the file pathname of any saved messages. This can be used as a default prompt for future saves and as a source of additional useful keywords.

8.5.3 Patterns

Pattern	Pattern Type
	evidence

The second top level component of the user model is a list of *patterns*. Patterns are created by agents and represent information deduced from the data kept in sessions. Lisp functions translate session data between CLOS objects (used by the INFOSCOPE interface) and OPS5 working memory elements (used by agents). New keyword terms and existing patterns are added to working memory. Agent rules scan session information for terms that match previously read terms and establish patterns for these repeated interests. Agents also scan for terms that support existing patterns. These patterns include keyword items that consistently appear, configurations commonly used and newsgroups commonly browsed. Patterns that persist across many sessions eventually lead to suggestions that the user create/modify a virtual newsgroup or create/modify a configuration. When a pattern is recognized this fact is entered into the user model. Agents then monitor these facts to determine if they are solid enough to warrant posting as a suggestion to the user. If patterns conform to the parameters of agents for a particular user (i.e., the terms in the pattern are read in 20% of the last 10 sessions), then the suggestion is made. Patterns include a pattern type and evidence to support the creation of that pattern in the user model. Pattern types vary with the rules agents use to detect them. In addition to pattern types, patterns contain evidence supporting the pattern and future computation concerning that pattern. Evidence consists of derived session data or pointers to session objects.

Pattern Type
group use
configuration use
newsgroup use
term deletion
term reading
term saving
frequency
recency
session duration

Examples of pattern types include patterns of group use, term reading and session duration. Agents designed to detect the frequency, and recency [Anderson 1991] of

patterns create pattern elements in the user model. This makes the patterns persistent so that they can be updated with respect to new user interests.

8.5.4 Extracting Information for Use by Agents

Initial priming (seeding) [Fischer et al. 1992] of the user model is performed by scanning the user's newsrc file. This provides INFOSCOPE with a list of subscribed to newsgroups and indicates which newsgroups have been omitted from the file by choice of the user. However, as mentioned earlier, most newsrc files are not good indicators of what newsgroups are actually being read because they do not contain information about the last time individual newsgroups were accessed. Newsgroups that have not been browsed for years may still show up as subscribed newsgroups. For this reason, INFOSCOPE does not place great weight on this information. It is useful for making early suggestions about configurations since it supports patterns of current newsgroup browsing. In addition, a program could be written to prime the model with patterns culled from previously stored messages, but this was not done for this project.

Data collection is accomplished by placing acquisition hooks into appropriate places in the INFOSCOPE code. For example, when a user saves a message or conversation to a file, the *useful message data* is stored. Similar acquisition is carried out whenever users create configurations or do anything else of interest. This is how a session object is populated during the execution of a session. When INFOSCOPE is exited, the session data is either written to a file or automatically mailed to a data collection site. LISP functions process the raw session data into a form usable by agent rules. For example, one function uses a *stop list* [Fox 1989] to filter out semantically meaningless terms from subject lines so that words like "and," "or," "if" etc. are ignored by pattern generating agents. In addition, certain message header lines will also be filtered out since they do not contain useful information. Agents then use this data to populate the user model with patterns. When patterns conform to agent parameters, they are instantiated as suggestions in INFOSCOPE.

One role filled by agents is the recognition of patterns already described. Rule sets scan the user model for sessions that exhibit patterns. These patterns are then placed back into the user model. Another role filled by agents is that of making suggestions. When patterns pass a threshold (by being around a long time etc.) they are instantiated as suggestions to the user.

8.6 Agent Architecture

The implementation and analysis of agents is a central focus of this research project. As previously discussed, this difficult task of reorganizing the information space requires assistance. The agents in INFOSCOPE are a collection of rule-based heuristics that utilize user models to facilitate the creation of suggestions to the user. These suggestions concern the future modification of the system model (e.g., suggestions for creating new structure or modifying existing virtual structure). Users can then accept, reject or modify these suggestions and evolution proceeds in this cooperative fashion (see Figure 8.2). Also, the agents analyze suggestions that are rejected or modified so that the user models are appropriately modified. When a suggestion is rejected,

something should be done to ensure that the same error in judgment is not included in future suggestions that occur due to similar circumstances (see Figure 8.2). For this reason users are provided access to the parameters agents use to determine when suggestions should be made (see Chapter 7). These parameters do not modify *what* the agents do, but why and when they do it.

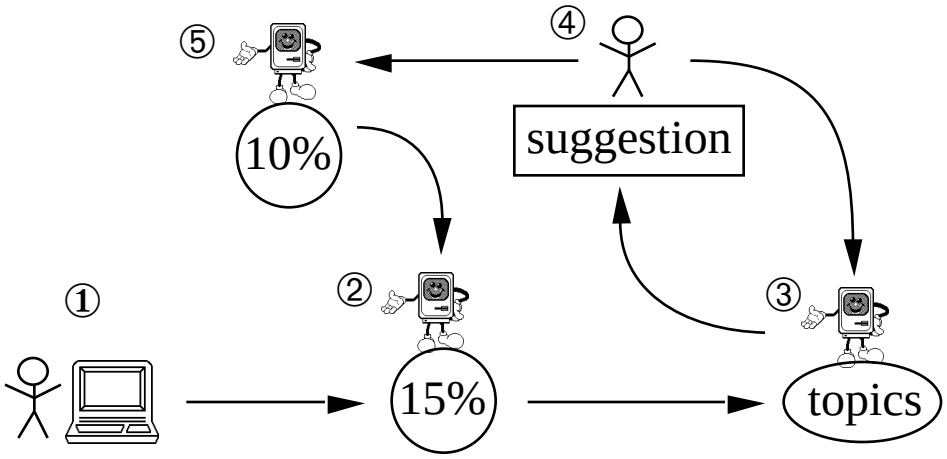


Figure 8.2 A Group of Agents

At ①, the user interacts with INFOSCOPE for a period of time that extends until agent ② recognizes that the user consistently reads less than 15% of the available messages in a particular newsgroup. At this time agent ② tells agent ③ to begin gathering topics judged interesting and uninteresting by the user. When a sufficient volume of data has been gathered agent ③ posts a suggestion to the user. (④) If the user accepts the suggestion agent ③ continues gathering topics for modifying filters later. If the user rejects the suggestion, the agent responsible presents its parameters for criticism by the user and uses this criticism to modify itself ⑤. In this example the user complains that the agent is too intrusive. It is modified so that it only reacts when the user reads 10% or less of the messages in a newsgroup.

It has been suggested that the introduction of agents into computer systems can be harmful [Wroblewski, McCandless, Hill 1989]. Especially agents that attempt to converse with users. This is because unlike human agents, computer agents do not have a shared understanding that can be utilized to model the knowledge state and goals of their human partners. Without this shared understanding, mixed–initiative dialogs cannot take place because the techniques we have for implementing such conversations are too mechanistic. This can lead to a breakdown in communication that leaves the human feeling stereotyped. INFOSCOPE agents avoid this pitfall by not attempting to converse with the user. Instead of implementing fully autonomous agents, I designed agents that construct filters. From the user's point of view INFOSCOPE agents serve as a feedback mechanism, helping them modify the displayed structure in response to observations of their own system usage patterns. Agents post suggestions to the user that must be accepted for them to be implemented. They do not converse with the users. In this model, shared knowledge about the information space is more important than a shared understanding of goals. An important result of this approach to agency is that the media of interaction are actually artifacts (the filter suggestions constructed by agents), rather than the utterances that dominate interactions in other types of agency models [Guindon 1988; Hill, Miller 1988].

8.6.1 A Personal Assistant Metaphor

Agents in INFOSCOPE are designed around a personal assistant metaphor that demonstrates some important characteristics of INFOSCOPE agents. Agents, like assistants, develop their models over time. In the beginning, agents do not know much about individual users. Suggestions may not be complete and poor suggestions are made. As time goes on, however, agents (like assistants) develop more specific models of their users and the actions they take are of a higher quality. In INFOSCOPE, users must accept the fact that agents may make bad suggestions. This research hypothesizes that this price is gladly paid to gain the advantages of the correct suggestions that agents will make. Use of the AGENDA system (see Chapter 5.4) has shown that users are willing to accept mistakes in automatic classification for similar reasons.

8.6.2 The Agent Suite

There are several agents that divide the task of helping the user into logical sub–tasks. For example, there are agents watching for conditions that dictate the creation of new newsgroups, an agent watching for new configurations etc. Users do not realize that they are interacting with different agents. This is true because all agents express themselves to the user in the exact same manner, by posting suggestions. Users have no way to know that there are really several different agents carrying out this task.

8.6.3 Agent Knowledge Structures

An agent is a set of heuristics (implemented as rules) that define knowledge about how to use the information, semantics and statistics kept in INFOSCOPE objects (where objects are groups, messages, the user model, etc.). The data in INFOSCOPE objects is also part of whatever agent utilizes it. Data is shared among many different agents without being replicated (really the standard object oriented approach).

The rules that create new virtual newsgroups, for example, get their information from various sources. Statistics kept in the objects representing each newsgroup supply this agent with details of how often the newsgroup is browsed, what type of messages are read and ignored (i.e., simple semantics like keywords and keyword pairs), and what type of messages are inherited from this newsgroup and displayed in others. The agent also gets other information from the user model. Some users may prefer many newsgroups with highly refined contents while others might prefer coarse filters on newsgroups that lead to diverse groups of messages. Still, other users may choose not to utilize the newsgroup creation facilities, and this information is located in the user model. The user model, in turn, may be modified by an agent consisting of rules that watch for shifts in user preference patterns. Observations made by this type of agent lead to suggestions to the user that evolve the user model.

The main metric used for determining which messages are interesting is matching keywords. These keywords are gathered in two major ways. Many of the keywords that determine (un)interesting messages may be supplied directly by the user through access to the user model. When suggestions are made, users will have the opportunity to modify the suggestion. This entails removing some bad keywords and/or adding

some more important ones. In this sense the INFOSCOPE system counts on a cooperative attitude from users. In breakdown situations, agents need help from users in finding the breakdown. Other keywords are gathered by an agent that scans messages for semantically important words. Words from read messages contribute to a list of interest indicators while words from unread messages contribute to a list of disinterest indicators.

8.6.4 Agent Creation

All agents have been created a priori as the system was developed. While there are no facilities for allowing users to create their own agents, the agents supplied with the system are somewhat user modifiable and are affected by criticism supplied by users. This does not exclude the possibility of adding user created agents later. There are many useful agents that could be implemented. This project concentrates on agents that help restructure the information space and help the users manage the newsgroup display graph.

8.6.5 Agent Adjustment

An agent's characteristics can change over time. Agent parameters are available to the user for editing whenever suggestions are made. Its abilities, however, do not change over time. Since adding abilities requires the addition of a robust set of heuristics/rules, allowing the user to do so at will is a difficult prospect [Fischer, Girgensohn 1990]. Editing agent characteristics occurs in two ways. The most common method is part of rejecting a suggestion. If a suggestion is rejected on the basis that it's information was insufficient, that might indicate this agent should wait longer before deciding it has useful information. This agent would become less intrusive in the future, allowing itself to gather better information before formulating a suggestion. The second method of agent modification is direct access to the agents. Modification in this manner will be just like modifying a filter definition. An agent will be selected (by clicking on a suggestion made by that agent) and it's parameters displayed. Users then change whatever dissatisfies them about particular agent characteristics.

8.7 Agent Implementation

As previously mentioned, agents are implemented as a set of OPS5 production rules. There are three main domains of rules in INFOSCOPE. First are the rules that scan session data extracting useful information for the potential creation of suggestions. The second domain of rules is the creation of suggestions for defining and evolving newsgroup configurations. The third domain is the creation of suggestions for defining and evolving virtual newsgroup filters. This section discusses the implementation strategy used to analyze session data and place suggestions back into the INFOSCOPE system. The following sections describe some representative rules and discuss the methods by which their deductions are made. In addition, there is a short discussion of the feedback mechanism by which users can alter agent parameters.

Agents in INFOSCOPE are not integrated with the news reading interface. Session data is saved to a file and processed by agents off line. This decision was made for several

reasons. The process of running rules over the session data is time and CPU intensive. In real working environments people do not have the time to wait ten or twenty minutes for agents to run each time they end a session, and the pattern generating process significantly reduces the performance of the machine running it. In addition, due to peculiarities in the implementation of OPS5, it is difficult to integrate the rule system into the existing interface. Finally, because agents are experimental it was necessary to actually watch the pattern generating process to ensure that errors in the rule base were not promulgating errors throughout the OPS5 working memory.

After the rules run over the available session data there may or may not be suggestions to offer to individual users. If there are any suggestions, a file is created and must be placed in the INFOSCOPE folder on the appropriate user's machine. This is all that is necessary to install the suggestions into the interface. The format of this file is essentially the same as the data file that contains the user's existing virtual newsgroup and configuration definitions. A filter definition consists of a newsgroup name, a list of parent newsgroups, a list of header field terms to include in the virtual newsgroup and a list of header field terms to exclude from the virtual newsgroup (see Chapter 7). If the suggestion is accepted (with or without modification) the final filter specification is then transferred into the file that contains installed filter definitions. The same process is carried out for the creation and installation of configuration suggestions.

In order to analyze session data, create patterns, or create suggestions working memory must be populated with data types (see Figure 8.3). These data types are used to control the order in which agents process data, to hold specific data elements and to test for circumstances that dictate various actions.

```
(literalize task              ; for controlling agents
      task_name               ; create_patterns (or)
                              ; suggest_configs (or)
                              ; suggest_newsgroups
      status)                 ; pending, active, finished

(literalize term              ; potential pattern element
      type                    ; existing, new
      keyword                 ; what was read
      field                   ; header field
      newsgroup               ; where it was read
      user_supported          ; t if user explicitly interested
      session_number)         ; for time based patterns

(literalize group             ; potential pattern element
      type                    ; existing, new
      name                    ; what group
      in_cofig                ; none or configuration name
      session_number)         ; for time based patterns

(literalize pattern           ; interesting items
      type                    ; term, group...
      keyword                 ; keyword if needed
      field                   ; header field if needed
      newsgroup               ; the group
      session_list            ; pattern trace
      strength_adjustment)    ; user initiated? (+), stale? (-)
```

Figure 8.3 Agent Data Types

Several OPS5 data types are defined using the literalize construct, enabling agents to keep track of information that leads to suggestions for new newsgroups and/or configurations.

8.7.1 Creating Patterns

The first step that agents take in analyzing session data is to search the data for terms that have appeared before. If a term has only appeared once before it will appear as an individual object in working memory. The first rules to run match each of the terms in the new session against terms that exist in this form. Any matches are then instantiated as patterns (see Figure 8.4). The creation of configuration patterns is handled similarly.

```
(p create_term_pattern
      (task ^task_name create_patterns ^status active)
      (term ^type existing ^keyword <word> ^field <field>
            ^newsgroup <group> ^session_number <session1>)
      (term ^type new ^keyword <word> ^field <field>
            ^newsgroup <group> ^session_number <session2>)
-->
      (make pattern ^type term ^keyword <word> ^field <field>
                    ^newsgroup <group>)
      (remove 2 3))

(p support_term_pattern
      (task ^task_name create_patterns ^status active)
      (pattern ^type term ^keyword <word>
               ^field <field> ^newsgroup <group>)
      (term ^type new ^keyword <word> ^field <field>
            ^newsgroup <group> ^session_number <session>)
-->
      (modify 2 ^session_list <session>)
      (remove 3))
```

Figure 8.4 Rules for Creating Patterns

The create_term_pattern rule creates a pattern for any word/newsgroup/field set that matches both a new and existing term object. The support_term_pattern rule adds a new session to the session list of an existing term pattern.

After new terms are checked against existing terms in working memory, agents proceed to check new terms against existing patterns. In cases where users have continued reading messages with the same terms, the existing patterns are modified to include the new session number. When terms appear multiple times in the same session, this information is recorded and triggers other agent rules that also support the existing pattern. This information is used by INFOSCOPE to determine if agent parameters have been met.

8.7.2 Suggesting Configurations

Suggestions are constructed partially by agents and partially by the INFOSCOPE component. This is because certain types of calculations are more easily implemented in Lisp than they are in OPS5. When patterns are present in working memory they are each written to a file. Patterns may or may not be complete suggestions. If some pattern is supported by additional patterns, they are all written to the file and together constitute the complete suggestion. When INFOSCOPE is launched, these patterns are loaded into the system and checked to see if they satisfy the current agent parameters. The patterns that satisfy the parameters are included in the final suggestion and presented to the user. If no patterns satisfy the constraints then no suggestion is made. These patterns remain in the user model for analysis by agents in the next cycle.

8.7.3 Suggesting Newsgroups

Newsgroup patterns are handled essentially the same as configurations (see Figure 8.5). As demonstrated in Chapter 8.7.1 agents only carry partial responsibility for

making suggestions. When patterns are created and supported (see Figure 8.4) working memory is populated with potential candidates for suggestion. The process of actually suggesting that a virtual newsgroup be created consists of writing these patterns to a file and submitting them to INFOSCOPE for testing against the parameters set for suggestion thresholds. Multiple patterns that support the same words being read on a consistent basis will end up in suggestions. Additional rules also cover the case where the same words are read from different sources. These sources are combined into a single suggestion with multiple parents.

```
(p suggest_configuration
      (task ^task_name suggest_configs ^status active)
      (pattern ^type group ^newsgroup <group>
               ^session_list <sessions> ^strength_adjustment <adj>)
-->
      (openfile pattern_file |new patterns| out)
      (write pattern_file (crlf) pattern <group> <sessions> <adj>))
```

Figure 8.5 Rules for Suggesting Configurations

Configurations are suggested when individual or sets of newsgroups are consistently browsed in INFOSCOPE sessions. Rules write all newsgroup patterns to a file and INFOSCOPE combines them into a completed suggestion.

9. Evaluating InfoScope

The evaluation of INFOSCOPE proved to be a challenging task. Lab studies could not supply sufficient information for assessing the success or failure of the system and the concepts it instantiates, mainly because laboratory studies are not designed to study how systems are used in realistic working conditions [Thomas, Kellogg 1989]. This realism was crucial to the evaluation of INFOSCOPE because agents must monitor actual user actions over extended periods of time in order to develop a model of interests. Several lab sessions in artificial conditions will not elicit the normal reading patterns required to observe this process. Therefore, two separate studies that play different roles in the overall assessment of INFOSCOPE were conducted. A short term laboratory evaluation of the INFOSCOPE system assessed the *usability* of the interface, supplied feedback for fixing interface problems before further study, and evaluated users' ratings of INFOSCOPE compared to the news reader they normally use. However, INFOSCOPE agents could not be evaluated. Therefore, a longer term naturalistic study (using INFOSCOPE as a test bed) was carried out to test the *usefulness* of INFOSCOPE. This included configurations, virtual newsgroups, and agent suggestions. Both studies were designed to elicit qualitative information about the success of the strategies employed in INFOSCOPE.

9.1 Lab Studies

The laboratory study was designed to confirm the usability of INFOSCOPE by observing subjects carrying out several tasks and comparing their overall impression of INFOSCOPE to the news reader they normally use.

Ten subjects participated in the short term evaluation. Each of them was an experienced Usenet news reader and they all used RN for reading news. Each subject was given a preliminary questionnaire asking about their level of experience and various questions about their typical news reading habits. Users were then given a tutorial and allowed to play with the interface until they were comfortable with it. After the tutorial session users were asked to carry out a set of predefined tasks. These included tasks like finding your favorite newsgroup, reading a couple of messages, saving a conversation, defining a virtual newsgroup, and replying to a message that I had posted. Users were asked to talk out loud as they were working and notes about interactions and problems were taken. Subsequent to the directed tasks, users were asked to engage in undirected browsing of the information space. Finally, users were asked to answer a second questionnaire after the session with INFOSCOPE. This questionnaire asked about the virtual newsgroup mechanism, conversation support and comparisons to other news reading software.

Results of the short term study were positive. Eight of the subjects rated the INFOSCOPE interface higher than the one they normally used (see Figure 9.1). While subjects did find fault with INFOSCOPE, only one user rated an existing news reader as preferable to INFOSCOPE (because INFOSCOPE ran too slowly). Subjects had some troubles using the virtual newsgroup mechanism but all of them indicated that they

would use such a mechanism if it were available in their news reader. Their input lead to changes in the interface before the long term studies. The agent mechanism was not tested in this study because no user model data could be gathered about the subjects prior to the evaluation.

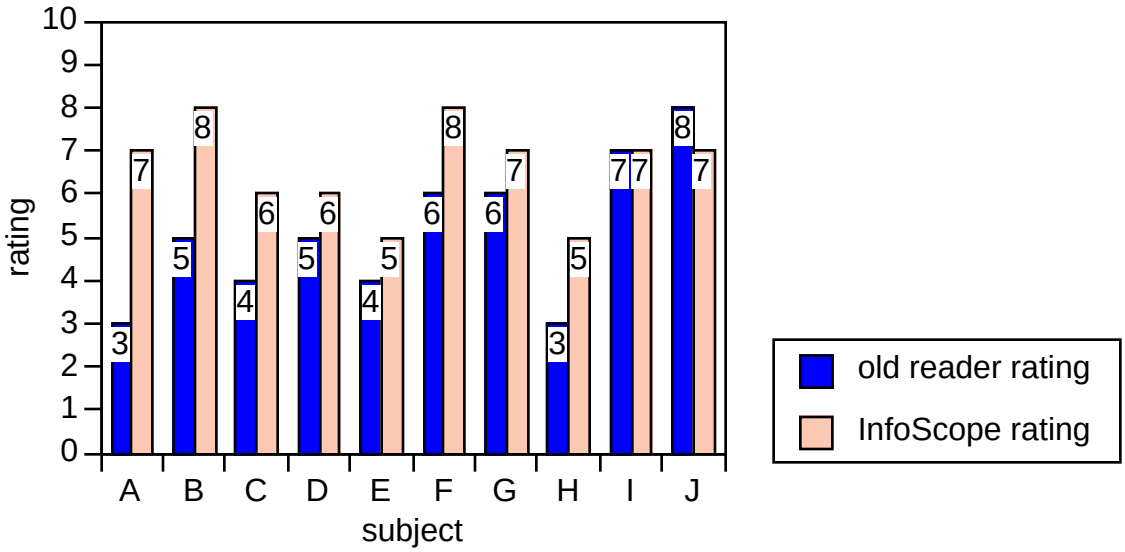


Figure 9.1 Rating RN vs. InfoScope

9.2 Naturalistic Studies

The long term evaluation was a naturalistic study in which 5 subjects used INFOSCOPE in their native working environments for extended periods of time (subjects participated for 10 weeks). They used INFOSCOPE anywhere from three to twenty times a week during this time. Data was gathered by the following mechanisms. Each participant filled out a questionnaire prior to using INFOSCOPE for the first time. Detailed information of their session interactions was automatically saved to a file that was either sent to me via electronic mail or made available to me over a network. In addition, INFOSCOPE requested that subjects rate any suggestions given (see Figure 9.2) as well as periodically requesting that subjects complete an on-line questionnaire about the levels of overload they are experiencing (see Figure 9.3). Occasional interviews were also conducted over the phone. Finally, each subject was asked to complete another questionnaire after the evaluation period was over.

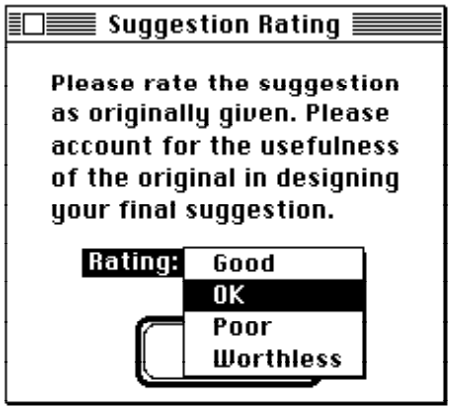


Figure 9.2 Rating Suggestions

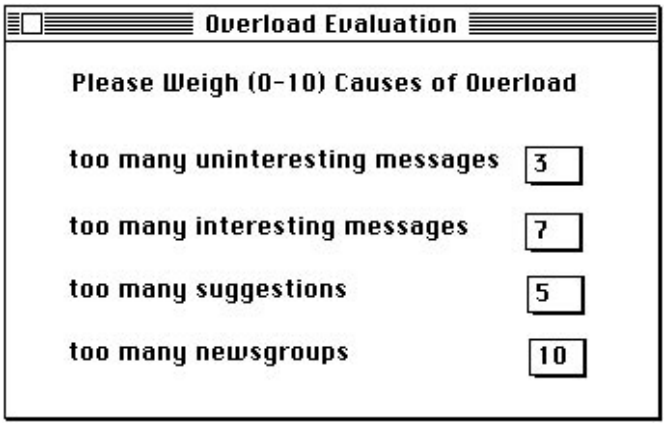


Figure 9.3 Evaluating Overload

9.2.1 Subjects

The five subjects chosen for this study read about a broad range of topics and used Usenet News for a variety of purposes. Subject A is a graduate student in electrical engineering. He reads news daily and browses both technical and recreational newsgroups on a regular basis. This subject subscribed to 23 newsgroups but only read from 14 of those during this study. Subject B is the president of a small consulting firm. He usually reads news about 3 times a week, but consistently reads the same newsgroups thoroughly. This subject subscribed to 17 newsgroups but read from only 8 during this study. This is the subject that successfully used a suggested filter to locate and purchase some Macintosh hardware (see Chapter 7.4). Subject C is an expert UNIX programmer. He reads almost exclusively from technical newsgroups related to his work, and his job description includes scanning several newsgroups for updated versions of software that is installed on numerous machines. This user subscribed to 60 newsgroups and read regularly from only 6 of those groups. Subject D is a secretary and while she reads news regularly, she only reads a few messages each day. On some of those days she is reading messages for her boss and not for herself. At these times she created virtual newsgroups to use as one–shot queries in order to search for information requested by her boss. She was originally subscribed to 4 newsgroups and read from 5 groups during this study. Subject E is a manager in a computer consulting firm. He reads news daily and was the highest volume news

reader participating in this study. This subject initially subscribed to 34 newsgroups and read from 12 of these during this study.

With only five subjects participating in the naturalistic study statistically significant results can only be claimed for those phenomena exhibited by all participants. While this is a limiting factor in the ability to claim significance for a wide range of results there were nevertheless several important findings that are significant.

9.2.2 Message Reading Levels

The first hypothesis explored in the naturalistic study is the assertion that INFOSCOPE use has an effect on the level of message reading activity. The specific hypothesis tested is that using INFOSCOPE has no effect on messages reading levels. In order to test this hypothesis with the small sample available a sign test was conducted. The sign test is a test based on the binomial distribution, where the test statistic is the number of positive differences in the sample [Wonnacott, Wonnacott 1977]. To obtain difference statistics for each subject, the reading level data for each subject was separated into the median levels for the first five weeks of the study and the median levels for the last five weeks of the study. The differences between reading levels at the beginning and end of the study were then tested.

An examination of the summary data (see Table 9.1) shows that subjects in the study increased their median message reading levels anywhere from 8 to 32 messages per week (subjects usually read news five days per week). While this is not a dramatic increase in the number of messages read, all five of the subjects did record an increase and by the sign test have rejected the null hypothesis that INFOSCOPE had no effect on message reading levels. The probability that these results are due to chance occurances is only .0313 at the 5% level.

Table 9.1 Effect of InfoScope on Message Reading

SUBJECT	Median Messages/Week (weeks 1–5)	Median Messages/Week (weeks 6–10)	Effect of InfoScope on Message Reading Levels
A	60	92	+32
B	124	132	+8
C	128	140	+12
D	25	48	+23
E	137	168	+31

Interviews with the subjects indicate that the low levels of message reading increase can be attributed to several effects. The most commonly mentioned reason was that users only allot a specific amount of time for news reading each day. These levels are not strictly adhered to, but users were trying not to let the INFOSCOPE study take away from their normal working time. Two users indicated that they were able to maintain their message reading levels while spending less overall time reading news. This was viewed as a significant improvement by those users. Two other users indicated that INFOSCOPE actually increased the time they spent reading messages as the number of interesting messages increased. This result is consistent with the findings of [Whitney

1978] that show the level of interest in an information item will directly effect the time devoted to examining that item. In the eighth week of the study subject A explicitly stated that he reduced message reading levels because he was spending too much time reading messages. Subject A made a similar comment after week seven of the study. These subjects did not say that they viewed this effect as a negative one, but comments like, “I wish I had more time to read” were commonly made. In addition, all five subjects indicated that reading e–mail was a higher priority task than reading news and the high volume of e–mail messages impacted the amount of news reading they could do.

Weekly median message reading levels for each subject can be seen in Figure 9.4. These graphs demonstrate that three of the subjects were more erratic in the amount of messages they read and they gave different explanations for these patterns. Subject A is a regular but relatively light reader of Usenet news who cuts back on reading when he spends too much time. Subject B read from groups that supplied differing levels of worthwhile information and subject E varied reading levels with the amount of recreational information he had the time to peruse.

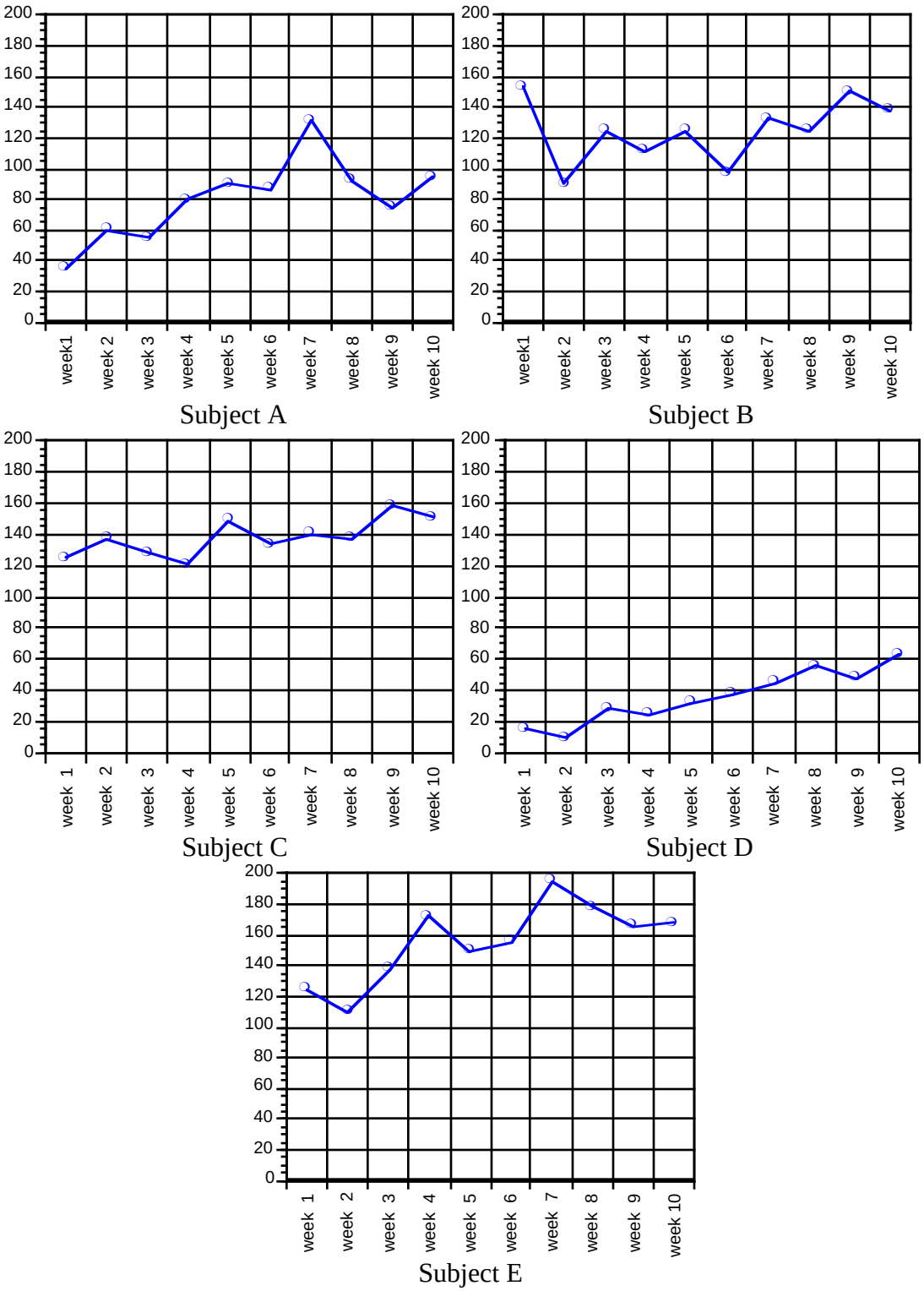


Figure 9.4 Increase in Message Reading Levels

9.2.3 Levels of Virtual Newsgroup Use

The next hypothesis tested is that INFOSCOPE users would not take advantage of the virtual newsgroup mechanism. This is an important test because virtual newsgroups are the main mechanism provided for reducing information overload. If virtual newsgroups were not used regularly, claiming a reduction in information overload due to that technology would be difficult. Difference statistics for a sign test were gathered in much the same way as for the previous test. The level of virtual newsgroup usage (measured as a percentage of the messages read that came from virtual newsgroups) was collected for each subject and medians for the first and second halves of the study were calculated. The differences between usage levels in these two halves of the study were then tested.

An examination of the summary data (see Table 9.2) shows that an increase in virtual newsgroup usage ranged from 11 to 52 perrcent between medians of the first and last five weeks. All five subjects exhibited an increase in the percentage of read messages that came from virtual newsgroups. By the sign test, the hypothesis that INFOSCOPE users would not utilize the virtual newsgroup mechanism can be rejected (p= .0313).

Table 9.2 Virtual Newsgroup Usage Levels Messages

SUBJECT	Median Usage (%) weeks 1–5	Median Usage (%) weeks 6–10	Change in Virtual Newsgroup Usage
A	88	99	+11
B	60	98	+38
C	43	95	+52
D	74	98	+24
E	55	95	+40

Individual weekly median usage rates (see Figure 9.5) show that users quickly and steadily adopted virtual newsgroups as their primary source of Usenet information. Subjects definitely increased their usage of virtual newsgroups as the study progressed. However, these results are mitigated by some of the interview results. Three of the users began reading from virtual newsgroups largely because they had agreed to participate in the study. Each of these users agreed that virtual newsgroups are very useful, but they might have adopted them more slowly had they acquired the software through normal channels. The other two subjects indicated that they had both been waiting for such capabilities and were quite anxious to use the mechanism.

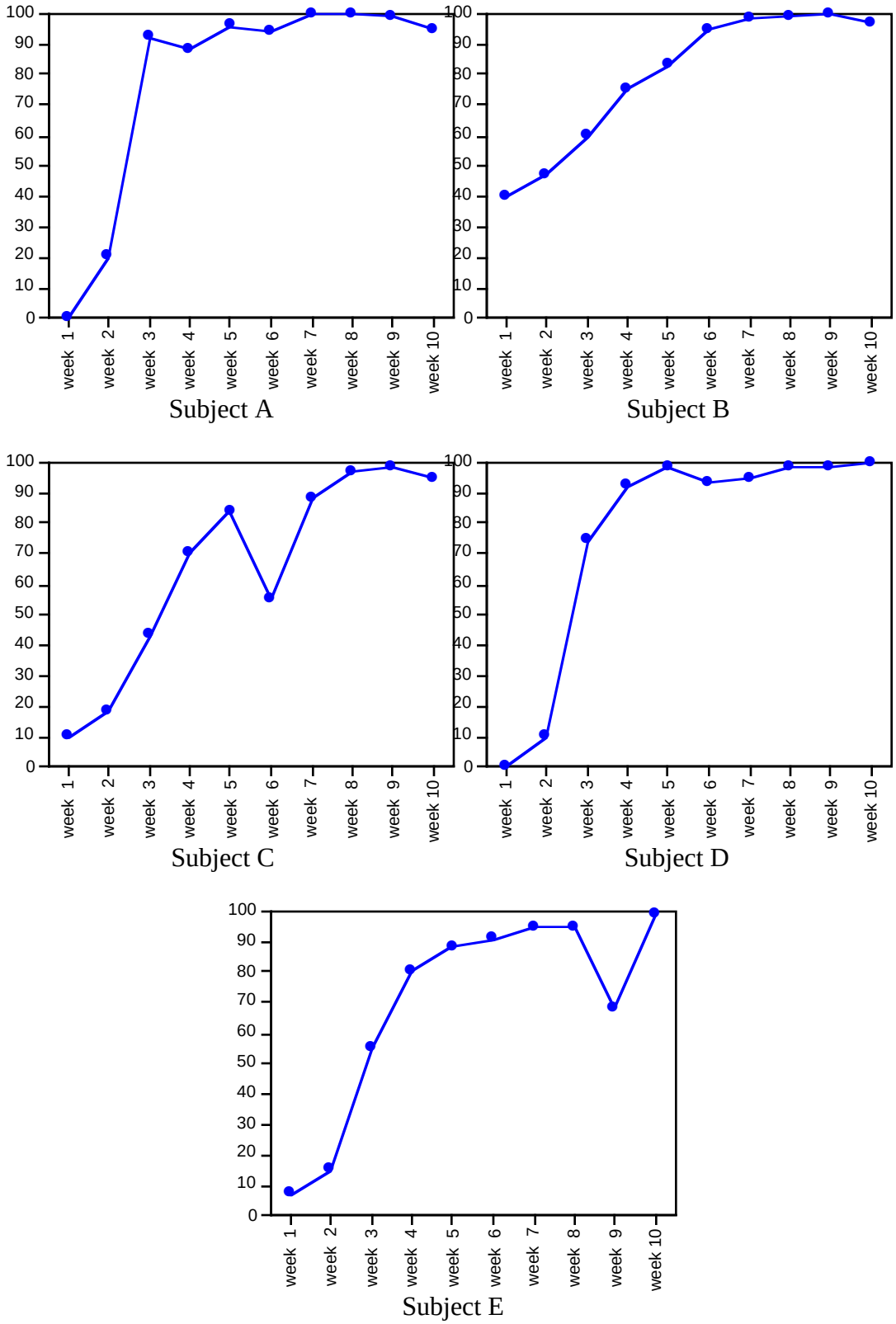


Figure 9.5 % Virtual Newsgroup Use

Post study interviews indicated that subjects were reducing their levels of vng usage, but that they were still utilizing the mechanism on a regular basis and still find it very useful Usage data after the official study period were not available because network connections used to gather data were no longer available.

9.2.4 Overload Due to Uninteresting Messages

These first two results were important because they were obtained through observation of INFOSCOPE users and did not rely on self evaluation by these users. However, users did supply self assessments (see Figure 9.3) and these were used to calculate the following results.

The hypothesis tested in this section is that INFOSCOPE does not help to reduce the information overload due to exposure to uninteresting messages. Sign test data were gathered by calculating mean overload assessments for each user over the first and last five weeks of the study. The differences between overload levels during these two periods were then tested.

An examination of the summary data (see Table 9.3) shows that subjects indeed rated overload levels in the second half of the study lower than they did in the first half. On a scale of one to ten, the decrease in overload due to uninteresting messages ranged from 1 to 3 points. Since all five participants in the study reduced their perceived overload, the sign test indicates that the hypothesis that overload levels are not affected can be rejected at the 5% level.

Table 9.3 Overload Levels Due to Uninteresting Messages

SUBJECT	Median Overload weeks 1–5	Median Overload weeks 6–10	Change in Uninteresting Messages Overload
A	5	4	-1
B	3	2	-1
C	6	3	-3
D	2	0	-2
E	5	2	-3

Interviews with the subjects showed differnt factors contributed to this overload evaluation. Subject D indicated that after one particularly relevant virtual newsgroup had been defined, she rarely felt that messages she was browsing were uninteresting. This did not mean that she was able to read all of the available information, only that she was perceiving less overload from messages she wouldn't like to read. This is reflected in the weekly overload levels (see Figure 9.6). Even when virtual newsgroup reading levels were very high, all subjects were forced by time limitations to ignore interesting messages in those groups. Some uninteresting messages appeared in virtual newsgroups, but subjects did not mention this as a significant source of overload during the interviews. Subject E experienced occasional rises in overload due to uninteresting messages and attributed this fact to browsing of non–filtered

newsgroups. However, this subject also indicated that this activity was “a gift to myself when I spent less time in my required reading.”

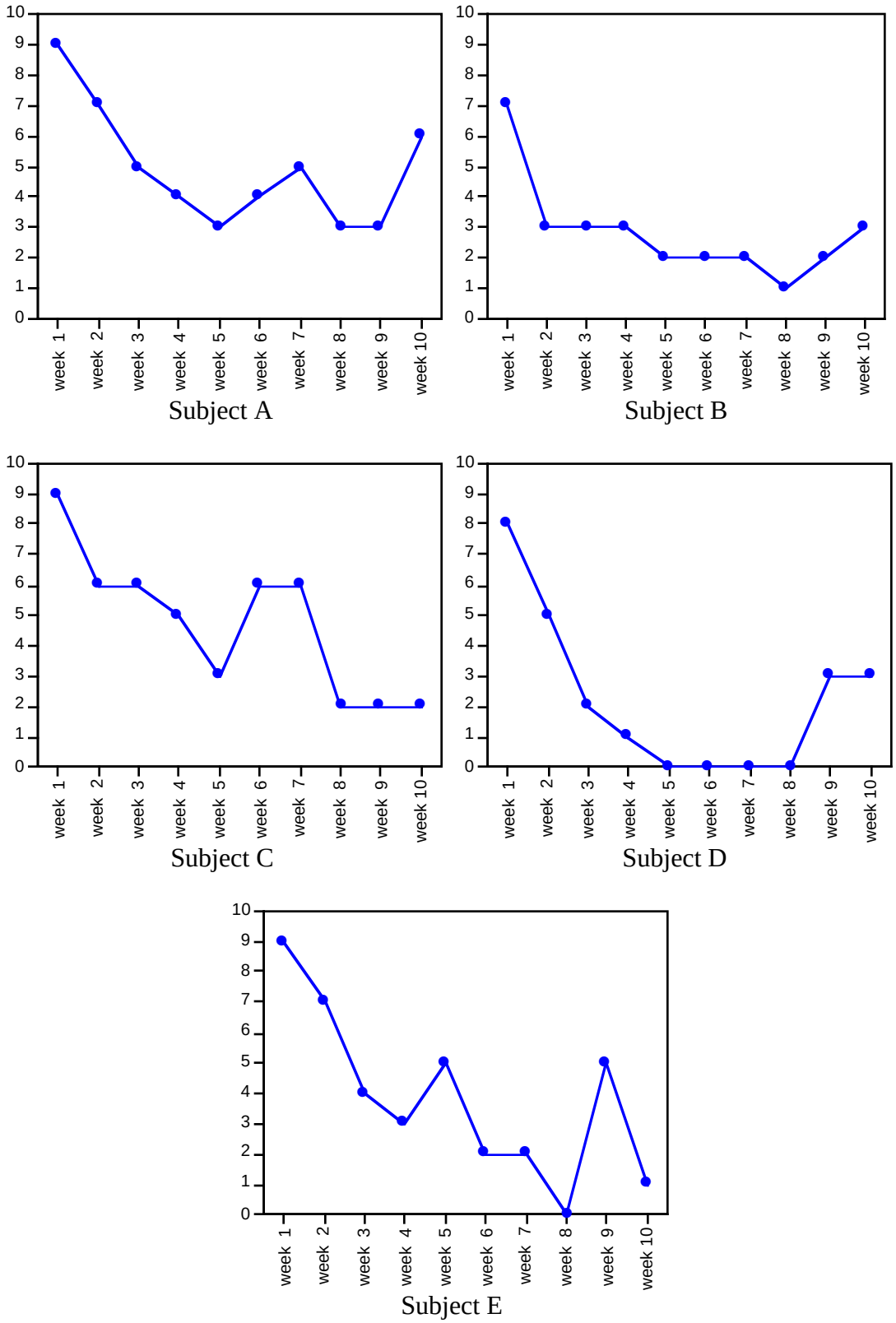


Figure 9.6 Uninteresting Message Overload

9.2.5 Configuration Use

The first time INFOSCOPE is launched each user is presented with the newsgroups that are subscribed to in their respective RN newsrc file. An empirical study conducted for this dissertation showed that RN users consistently subscribed to more newsgroups than they actually read. This result was reflected in the subjects of the naturalistic study. Users were initially presented with more newsgroups in the display than they were regularly reading. During this initial phase, before defining display configurations, users were consistently more overloaded than after they began utilizing the configuration features. These data are summarlized in Table 9.4. Once again, a sign test confirms that this reduction in overload due to too many newsgroups when using configurations is significant at the 5% level. All five users demonstrated some reduction in overload when displaying customized configurations.

Table 9.4 Overload Levels Due to Newsgroups

SUBJECT	Median Overload w/out Configuration	Median Overload with Configuration	Effect of Configurations on Newsgroup Overload
A	8	3	-5
B	5	3	-2
C	10	7	-3
D	3	0	-3
E	4	3	-1

9.3 Other Observations

INFOSCOPE usage data shows that all users experienced high levels of information overload in Usenet News. The main goal of this research was the reduction of effort in searching for information and the increase of time available to deal with interesting messages. The creation of configurations helped focus users attention on their most frequently used newsgroups and reduced information overload caused by too many newsgroups. As more virtual newsgroups were defined, the ratio of messages read to those ignored during INFOSCOPE sessions increased. The source of overload reported by users shifted from uninteresting messages to interesting messages, indicating more efficient use of their time.

9.3.1 Motivations for Creating Filters

Interviews with subjects indicated that their motivations for creating and modifying filters fell into four categories. The most common reason users created their own filters was in anticipation of some future need of information. All five subjects created filters in the first 24 hours for this reason and continued to occasionally modify these filters or create new ones as the study continued. This modification of filters due to a change in recognized interest patterns was the second category of motivation. This is consistent with the result of THE INFORMATION LENS studies [Mackay 1990b] that showed users were willing to create and modify filters for recognized interest patterns. INFOSCOPE agents were not able to help with these early filters. In the initial sessions agents did not have the time to create any patterns of user

interest. The priming of the user model is done from a newsrc file that contains information about what newsgroups users are subscribed to and not what message topics they are interested in. Until subjects demonstrated an actual interest in specific messages over a period of time, agents could not suggest filters on any specific topics. This is not surprising because agents are not designed as predictors of future interest, but as observers of past interest.

Another reason that users manually created filters was to satisfy a current and pressing need for specific information. These filters were essentially used as database queries, serving as one–shot requests for information that were never used again. Only one of the subjects used this strategy, but she used it frequently in response to requests for information from her boss.

The final category of motivation for creating filters corresponded to situations where users wanted to exclude messages from certain people or conversations on certain topics from their view. This occurred in three instances. The first was a case where the subject read a posting that no equipment should be purchased from a particular individual because of problems other people had with him. The second was a case where a large set of FLAMES were being posted in response to an inflammatory message. These two cases were from subject E. In the third case a subject C did not like someone who commonly posted messages to a newsgroup and defined a virtual newsgroup that contained all messages except those posted by that individual. These observations are interesting because these filters were essentially used in the same manner as KILL files in RN. None of these users, however, had ever used KILL files while using RN.

9.3.2 Evaluating Suggestions

While many suggestions for filter creation were heavily modified or rated as poor, approximately 50% (16/31) of the suggestions considered were used in some form (even if only as a cue that a filter of some kind was needed). 63% (10/16) of the suggestions that were accepted were rated as poor. These suggestions were all filter suggestions that were heavily modified, using an average of only 23% of the suggested terms. They were used, however, indicating that users are willing to expend the effort necessary to compensate for the knowledge poor techniques used for filter suggestion. Only 6% (1/16) of the suggestions were rated as excellent and 18% (3/16) were rated as good, indicating that users did not consider a heavily modified suggestions to be good suggestions, despite the fact that they were using these suggestions.

All of the suggestions rated in the good and excellent categories were configuration suggestions. This is not surprising because agents had little trouble computing what newsgroups were actually being read. All configuration suggestions (5/5) were accepted, and these suggestions required little modification. Subjects B, C, and E each created their own initial configuration. A subsequent suggestion to create another configuration was accepted by all three and they continued to have two configurations for the remainder of the sessions. Subject B, however, said he would have preferred an option to combine this two configurations into a single configuration. Subjects A and D created their first configuration in response to a

suggestion and continued using that configuration during the remaining sessions. All subjects indicated that configurations were necessary to reduce information overload due to too many displayed newsgroups.

Interviews with subjects indicated that the ratings of suggestions were directly related to the amount of time that was spent fixing them Since so many of the suggestions required intervention by the subjects they were inclined to rate the suggestions poorly. In addition, 66% (10/15) of the suggestions that were not accepted were considered but rejected. The other 5 suggestions were ignored completely due to time constraints or lack of a desire to consider any more suggestions. These observations were confirmed by a slightly increasing overload due to too many suggestions at the end of the study.

Almost no suggestions (2/16) were rated worthless. However, the 5 suggestions that were completely ignored could be considered rejected suggestions that subjects didn't want to take the time to deal with. Several suggestions were rated poor, but most of these were modified to create a useful filter. However, for the suggestions that were rated poor, almost all of the suggested terms were removed and replaced by alternate terms. This could mean that the one or two remaining items in the suggestion were effectively serving as cues for the subjects, but it also means that most of the terms in those suggestions were of no use.

9.3.3 Perceived Benefits from Personalization

Subjects were also interviewed about the benefits they perceived from expending effort to personalize their information space. This information is summarized in Table 9.5. The interviews showed that while their participation in the experiment did have some bearing on their willingness to explore suggestions, this was not mentioned as a motivation in the manual creation of virtual newsgroups.

Table 9.5 Why Users Spend Time On Personalization Tasks

Task	Benefit
manually create virtual newsgroups	• explore INFOSCOPE features • filter current interests • find information from boss • suggested by another user • can't wait for relevant suggestion
consider suggestions	• explore INFOSCOPE features • had spare time • test usefulness for the study • some suggestions were useful • tired of creating filters manaully • ran out of filtering ideas • fun

9.3.4 Example of a Failed Suggestions

It is useful to look at a suggestion that was rated as worthless because it points to one of the limitations of the techniques employed in INFOSCOPE. Subject C works for a consulting firm and his responsibilities include scanning Usenet for new software and updates/patches to existing software installed on client machines. This user saved a significant number of messages from newsgroups that contain only software. Some software packages are quite large and are distributed over as many as 20 messages. Because the user was saving these messages, the terms in them were being heavily weighted as indicating persistent interest in those topics. However, this user had selected the option that prevents terms from a single conversation from being included in filter suggestions. Therefore, the only terms that all of the messages had in common were in the From: header field. Agents noticed this pattern and constructed a suggestion that filters all messages from that person into a virtual newsgroup. The suggestion was considered by this subject and immediately rejected and given a rating of worthless.

Examination of the event and a subsequent interview showed a clear reason for this reaction. Newsgroups that distribute software are commonly moderated. This means that all messages are mailed to a single person whose job it is to decide what messages are suitable for posting There are several types of newsgroups that become moderated and the decision to do so is usually made during the voting period when the charter of new newsgroups is established. The reasons for moderating binaries newsgroups are twofold: (1) they are moderated to control the pace and quality of postings so that software distribution does not overwhelm network bandwidth, and (2)

they are moderated so that the moderator can test the software for viruses. The result of this is that all messages in a moderated newsgroup are posted by the same person. Therefore, defining a filter that collects all messages from that individual is a waste of time. The virtual newsgroup would contain exactly the same messages as the original newsgroup.

10. Limitations, and Future Work

When completing a research project the question of what one would do with several more years to work on the project always arises. This section discusses some of the limitations of this research and the steps that might be taken to address them.

10.1 Filters and Suggestions

Several steps should be taken to improve the filtering capabilities of INFOSCOPE. One of these stems from the fact that making suggestions to modify filters needs a new representation. In the current implementation these types of suggestions are made by presenting the new complete filter with the same name as the existing filter. This representation was confusing to users and all suggestions of this type were rejected. When interviewed about these suggestions one subject indicated that he rejected the suggestions "because a filter for those newsgroups already existed." One potential solution to this problem would be to represent these suggestions as differential descriptions rather than complete filters. Currently, users must perform the mapping between the existing filter and the new suggestion by manually inspecting both filters for differences. Differential descriptions make the suggested changes explicit and allow users to concentrate on the new or removed items. This is important if agents are going to be effective in keeping filters relevant to current interests. In the current system all of these kinds of modifications were made manually by users when they realized that uninteresting messages were being included in virtual newsgroups. The patterns of change that agents were able to suggest were lost in the confusion of examining and comparing multiple filter definitions.

Also, INFOSCOPE'S ability to define filters is limited to a small subset of Boolean logic. This means that some expressions are difficult to make. It is not even clear that Boolean logic is the best way of defining filters [Greene et al. 1990]. For example, a technique called Latent Semantic Indexing has proven better than keyword techniques for clustering messages in the news domain [Foltz 1990]. This technique encodes entire messages into a multi–dimensional space. Messages that cluster together are somehow related. A significant advantage of such a technique over Boolean matches is that keywords need not necessarily co–occur in two messages for them to be considered related. Unfortunately the process of creating the needed vector space is computationally expensive and this is why it was not used in this implementation.

Another drawback of the INFOSCOPE system is that it can't consider the content of messages. This is where the personal assistant metaphor fails to match INFOSCOPE. Currently the system works entirely in the header fields of a message. Since vocabulary in the subject line may not match the vocabulary of the message body, some relevant messages are missed by filters. Expanding the search for keywords to message bodies is another potential enhancement that could yield significant improvements. However, it is also possible that doing this would result in suggestions that are even more cluttered with irrelevant terms thereby causing users to expend even more effort in the modification of those filters.

A related problem is that INFOSCOPE agents have no semantic understanding of the purpose of different newsgroups. This was especially evident in the example of bad suggestions described in Chapter 9.3.4. By supplying agents with an understanding of different newsgroups types they could avoid making common errors. Newsgroups designated for the distribution of software should be treated differently than those designated for as discussion groups. A conversation in a binaries group corresponds to a complete software distribution and should be automatically combined into its native form. The current system only has support for discussion type newsgroups.

10.2 Enhancing Interactions With Agents

The current version of INFOSCOPE supplied users with a rudimentary method for supplying feedback to agents that are making poor suggestions. When a filter is rejected or the user wants to explicitly criticize the agent that made the suggestion, the user is provided with a dialog enabling them to adjust agent parameters (see Figure 10.1). On the left side of the dialog the parameters that were responsible for the suggestion are enumerated. On the right side of the dialog, users are afforded the opportunity to adjust these parameters. In addition, users may choose to have agents ignore some of these parameters.

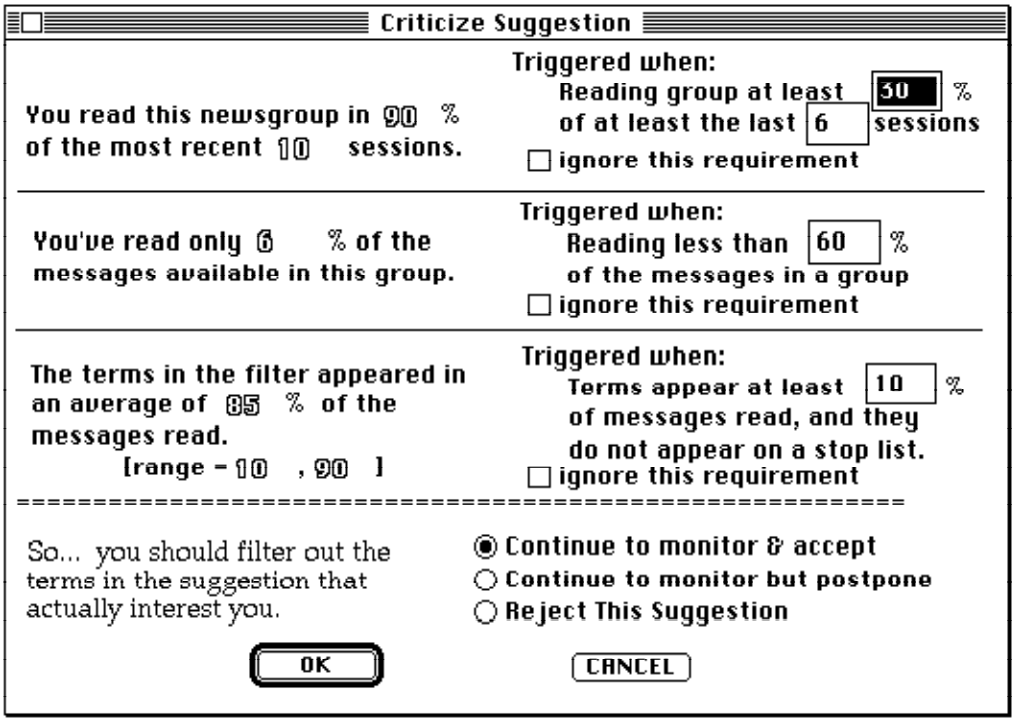


Figure 10.1 Criticizing Agents

There were two problems with this approach to agent adjustment. The first problem is that the interface was not making it clear that these adjustments did not apply to all agents, but only to the future calculation of the suggestion in question. Users were reluctant to change the parameters because they felt they were changing the global settings of agents. Interviews indicated that users were afraid of “ruining” the agents and not getting any future useful suggestions. This was not made clear enough in the

tutorial sessions and any future version of INFOSCOPE should make the effect of agent adjustments more clear.

The second problem with the INFOSCOPE agent adjustment mechanism was that users were not willing to take the trouble to use it. All subjects in the user studies either ignored this capability or chose not to use it in favor of adjusting the suboptimal filters the agents were suggesting. The feeling was that their time was better spent modifying filters to be useful than attempting to figure out what agent adjustments would create better filters. One reason for this may be that agents were using statistical mechanisms for tracking user interests. Since users were viewing their message interests in terms of the meaningful keywords that they recognized rather than their statistical properties, they found it difficult to translate those interests into statistical adjustments. Users were more comfortable adding and deleting terms because they were confident of the effects those types of adjustments would have on filtering. Future versions of INFOSCOPE may be able to mitigate these problems by showing the effects that individual agent adjustments would have had on suggestions that were already presented to users.

10.3 Supporting Groups With Catalogs

Evaluations of the INFORMATION LENS system have shown that users are willing to share information filters. INFOSCOPE should provide users with a mechanism for doing this. Catalogs of useful filters are one approach to this problem. Examples in the catalog should be accessible by presenting the system with messages, essentially satisfying the request, “Are there any filters in the catalog that would retrieve messages like this?”. This also points out a potential use for stereotype user models. If users can be categorized into effective stereotypes, then new members of a particular group could be presented with sets of filters that are commonly adopted by other users that fit into that stereotype.

10.4 User Interface Problems

During the design and short term evaluation of INFOSCOPE the interface went through numerous changes. The main problem with the current implementation is the mix of interface metaphors that were necessary due to the historical evolution of the system. Originally, INFOSCOPE was implemented on a SYMBOLICS machine, utilizing the presentation type mechanism provided by that environment. Using presentation types, objects on the screen are automatically highlighted whenever the mouse passes over them in appropriate contexts. If the system is awaiting command input then any object on the screen that can be interpreted or used as a command will be highlighted when the mouse passes over it. When INFOSCOPE was ported to the MACINTOSH environment this interaction style was maintained in the newsgroup and message browsers. Unfortunately, the native Mac interface does not provide this style of interaction. MACINTOSH users are used to an interaction style that requires them to explicitly select objects before acting on them. While users were eventually able to adjust their interactions to account for the difference, there were several complaints about accidental mouse clicks that produced unwanted results. A future version of INFOSCOPE should maintain the standard MACINTOSH interaction style, much as existing Mac news readers have done (see Figure 10.2).

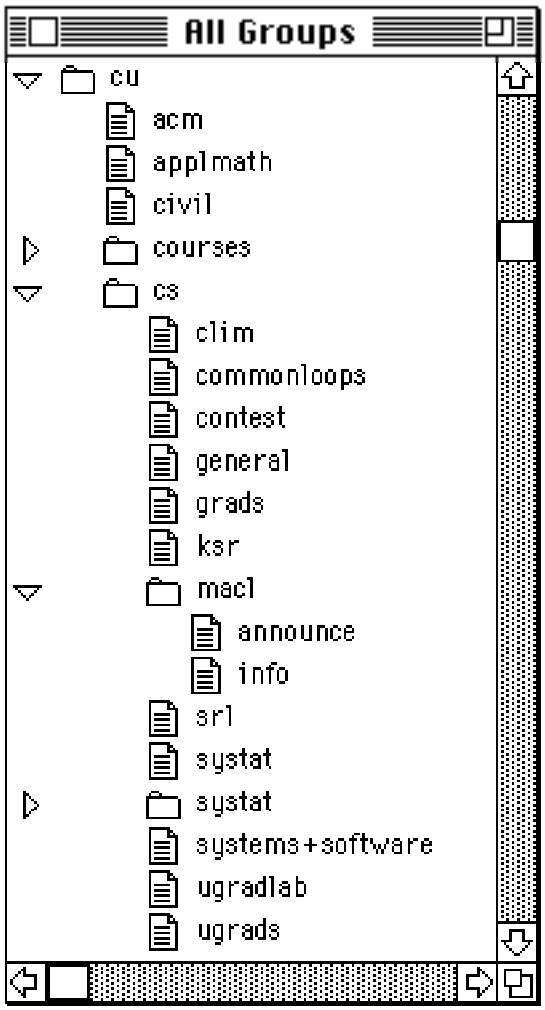


Figure 10.2 The Nuntius News Interface

11. Summary

Computers serve well as a medium for making more and more information available to their users. The critical resource is not the available information but the human attention necessary to utilize that information [Simon 1981]. This project is a combination of theoretical and operational research. The reality of information overload leads to the need for information filtering. This concept is realized in the INFOSCOPE system through the implementation of virtual newsgroups based on message filters. INFOSCOPE bridges the gap between the situation and system models by allowing users to define persistent virtual newsgroups that are organized by individually chosen semantics. Finally, the undesirable cost/benefit ratio of maintaining complex filters, and the existence of user interest patterns of which they are not aware, leads to the need for agents. With agents, INFOSCOPE becomes a cooperative problem solving system, allowing users to define their own extensions to the a priori structre using adaptable features while assisting users in this process through adaptive features. Utilizing these features users evolve their own personal view of an information space as interests change.

The central issue in this thesis is whether or not virtual newsgroups and agents are an effective mechanism for reducing information overload in Usenet news. This has been verified by the following:

- users defined filters for interests they were aware of and consistently preferred reading messages from those groups
- users utilized many of the agent suggestions despite the fact that most of those suggestions were rated as poor
- users read more messages at the end of the study when they were reading from virtual newsgroups than they were reading at the onset of the study without virtual newsgroups
- levels of overload due to uninteresting messages were lower at the end of the study when subjects were reading most of their messages from virtual newsgroups, however…
- levels of overload due to interesting messages were higher at the end of the study when subjects were reading most of their messages from virtual newsgroups
- a high percentage of messages that are filtered into virtual newsgroups are actually read
- users enjoy using INFOSCOPE, preferring it to their normal news reading interface

Since most suggestions were modified, more sophisticated agents are needed to reduce the extraneous terms in those suggestions. The evaluation of INFOSCOPE has illuminated several principles for the design of information agents:

- agents must be flexible
  even if they do not understand the content of individual messages, they must have an understanding of the semantics of the information sources
- agents must present adjustments that users can understand and manipulate
- adjustments must remain meaningful enough to allow users to balance filtering needs against overload from suggestions
- agents must allow users to criticize their suggestions

Messaging systems are becoming more and more popular. As people begin to prefer them to paper mediums and rely on them for more important tasks, it becomes crucial that users are able to manage their information effectively. In addition to issues mentioned above, INFOSCOPE has extended the current state of the art in information access technology by exhibiting the following advantages:

- users manage suggestions (in the situation model) instead of managing agents or the structure (in the system model) — agents assume much of the burden of mapping from the situation model from the system model
- the structure created by users can be used to augment personal information environments with the extra structure needed to place poorly structured information in those types of highly structured information spaces
- this system demonstrates that evolving system models, based upon principled examination of user situation models, is an effective way to reduce the overload problems associated with large, poorly structured information spaces
- agents were able to make useful suggestions using a minimal amount of knowledge about newsgroup semantics and message content

Furthermore, two of the virtual newsgroups that were defined during this study had something unique in common. Each of them was suggested to the Usenet community for inclusion in the a priori hierarchy. The requests to do so were not initiated by any of the subjects in this study, but each of them voted for the official creation of those groups. One of these newsgroups was suggested to a subject by an INFOSCOPE agent. Agents suggested that one subject create a newsgroup that contained specific Macintosh items that were being offered for sale. In the final week of the study, a newsgroup called misc.forsale.computers.Macintosh was actually voted on and created for the entire Usenet readership. While the virtual newsgroup suggested was more restrictive than any Macintosh product, it is interesting to note that so many users in the general community felt that this was a useful classification.

This research proposed approaches to information overload problems based on a principled conceptual framework. Each component of the INFOSCOPE system (views,

virtual newsgroups, agents, suggestions) is an approach to addressing one or more of these problems. Below is a table that maps between topics discussed in the conceptual framework of this dissertation, the goals & approaches of this research, and some evaluation results.

Topic	InfoScope Approach	Evaluation
Situation vs. system Model (sender's needs do not match reader's needs)	reduce effort required for finding information; provide mechanisms for readers to organize information around personal situations, give assistance based in situation models	users preferred reading messages from personalized newsgroups because of the improved incidence of relevant information
The Role of Structure	reduce effort required to browse a priori structure; provide mechanisms for adding personalized structure to the a priori structure	views were used to reduce the size of the information space; some personal structure was eventually adopted by the whole user community
Personalization	reduce effort required to customize environment; provide mechanisms by which the system shares responsibility for personalization with user	users were willing to accept and/or modify even poor suggestions in return for the effort reduction in finding interesting messages
Adaptive & Adaptable Systems	reduce effort required to instantiate personal reading patterns; provide mechanisms that recognize patterns & instantiate them	recognized patterns contained too much noise and were often used primarily as cues that some filtering was necessary; some users modified entire suggestions but did use them
Extending Critics	allow mixed–initiative interactions where user & system switch roles; provide agents that construct filters and allow users to critique them	users preferred manual creation of filters for patterns they recognize, but utilized suggestions for unrecognized patterns (after modifications) even when they rated them poorly
Information Overload (too many newsgroups)	allow users to ignore uninteresting newsgroups by eliminating them from display; provide a mechanism by which users can define personal views of Usenet hierarchy	all users defined at least one view (a default view) that contained a limited set of interesting newsgroups after which overload decreased; some users defined several views
Information Overload (due to too many uninteresting messages)	reduce overload by utilizing personalized filters; provide virtual newsgroup and agent (suggestion) mechanisms	significant reductions in overload due to noise were reported by all users, however, some of this overload transferred into overload due to too many interesting messages
Information Overload (due to too many interesting messages)	deliver messages that have a demonstrated interest to the user; agents recognize changes in interest patterns and keep filters up to date	as overload due to uninteresting messages decreased, this overload increased; users often ignored suggestions to remove terms from filters; message priority system should be developed; future work
Information Overload (due to too many suggestions)	keep number of suggestions high enough to be useful (and to test them) while not inundating users; reduce number of suggestions when they are ignored or consistently rejected	overload levels for suggestions varied by user; more sophisticated method of prioritization left for future work

12. References

[Anderson 1990]
J.R. Anderson, *The Adaptive Character of Thought*, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1990.

[Anderson 1991]
J.R. Anderson and L.J. Schooler, *Reflections of the Environment in Memory*, Psychological Science, Vol. 2, No. 6, November 1991, pp. 396–408.

[Baker 1985]
S.L. Baker, *An Exploration Into Factors Causing the Increased Circulation of Displayed Books*, Phd Dissertation Thesis, School of Library and Information Science, University of Illinois at Urbana–Champaign, 1985.

[Bergstrom, Stoll 1990]
J.C. Bergstrom and J.R. Stoll, *An Analysis of Information Overload with Implications for Survey Design Research*, Lesiure Sciences, Vol. 12, No. 1990, pp. 265–280.

[Bobrow et al. 1977]
D.G. Bobrow, R.M. Kaplan, M. Kay, D.A. Norman, H. Thompson and T. Winograd, *GUS, A Frame–Driven Dialog System*, Artificial Intelligence, Vol. 8, No. 1977, pp. 155–173.

[Bobrow, Bower 1969]
S.A. Bobrow and G.H. Bower, *Comprehension And Recall Of Sentences*, Journal of Experimental Psychology, Vol. 80, No. 3, 1969, pp. 455–461.

[Broadbent 1978]
D.E. Broadbent and M.H.P. Broadbent, *The Allocation of Descriptor Terms by Individuals in a Simulated Retrieval System*, Ergonomics, Vol. 21, No. 1978, pp. 343–354.

[Carroll, McKendree 1987]
J.M. Carroll and J. McKendree, *Interface Design Issues for Advice-Giving Expert Systems*, Communications of the ACM, Vol. 30, No. 1, January, 1987 1987, pp. 14-31.

[Carroll, Rosson 1987]
J.M. Carroll and M.B. Rosson, *Paradox of the Active User*, in J.M. Carroll (eds.), *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, The MIT Press, 1987, Ch. 5.

[Denning 1991]
P.J. Denning, *Short Citedness*, Communications of the ACM, Vol. 34, No. 5, May 1991, pp. 17–19.

[Dijk, Kintsch 1983]
T.A.v. Dijk and W. Kintsch, *Strategies of Discourse Comprehension*, Academic Press, New York, 1983.

[Fischer 1988]
G. Fischer, *Cooperative Problem Solving Systems*, Proceedings of the 1st Simposium Internacional de Inteligencia Artificial (Monterrey, Mexico), October 1988, pp. 127-132.

[Fischer 1990]
G. Fischer, *Communications Requirements for Cooperative Problem Solving Systems*, The International Journal of Information Systems (Special Issue on Knowledge Engineering), Vol. 15, No. 1, 1990, pp. 21-36.

[Fischer 1992]
G. Fischer, *Shared Knowledge in Cooperative Problem-Solving Systems - Integrating Adaptive and Adaptable Systems*, Proceedings of 3rd International Workshop on User Modeling (UM'92), The German Research Center for Artificial Intelligence, Dagstuhl, Germany, August, 1992 1992, pp. 148-161.

[Fischer, Girgensohn 1990]
G. Fischer and A. Girgensohn, *End-User Modifiability in Design Environments*, Human Factors in Computing Systems, CHI'90 Conference Proceedings, ACM, Seattle, WA, April 1990, pp. 183–191.

[Fischer et al. 1992]
G. Fischer, J. Grudin, A.C. Lemke, R. McCall, J. Ostwald, B.N. Reeves and F. Shipman, *Supporting Indirect, Collaborative Design with Integrated Knowledge-Based Design Environments*, Human Computer Interaction, Special Issue on Computer Supported Cooperative Work, Vol. 7, No. 3, 1992,

[Fischer, Henninger, Redmiles 1990]
G. Fischer, S. Henninger and D. Redmiles, *A Conceptual Framework and Innovative Systems for Accessing Knowledge for Software Reuse*, Proceedings of the HCIC Consortium 1990 Winter Conference, HCIC, San Diego, CA, February 1990,

[Fischer, Henninger, Redmiles 1991]
G. Fischer, S.R. Henninger and D.F. Redmiles, *Cognitive Tools for Locating and Comprehending Software Objects for Reuse*, Thirteenth International Conference on Software Engineering (Austin, TX), ACM, IEEE, Los Alamitos, CA, 1991, pp. 318-328.

[Fischer et al. 1991a]
G. Fischer, A.C. Lemke, T. Mastaglio and A. Morch, *The Role of Critiquing in Cooperative Problem Solving*, ACM Transactions on Information Systems, Vol. 9, No. 2, 1991a, pp. 123-151.

[Fischer et al. 1991b]

G. Fischer, A.C. Lemke, T. Mastaglio and A.I. Morch, *Critics: An Emerging Approach to Knowledge-Based Human Computer Interaction*, International Journal of Man-Machine Studies, Vol. 35, No. 5, 1991b, pp. 695-721.

[Fischer, Lemke, Schwab 1984]
G. Fischer, A.C. Lemke and T. Schwab, *Active Help Systems*, Readings on Cognitive Ergonomics - Mind and Computers, Proceedings of the 2nd European Conference (Gmunden, Austria), September 1984, pp. 116-131.

[Fischer, Lemke, Schwab 1985]
G. Fischer, A.C. Lemke and T. Schwab, *Knowledge-Based Help Systems*, Human Factors in Computing Systems, CHI'85 Conference Proceedings (San Francisco, CA), ACM, New York, April, 1985 1985, pp. 161-167.

[Fischer, Nakakoji 1991]
G. Fischer and K. Nakakoji, *Making Design Objects Relevant to the Task at Hand*, Proceedings of AAAI-91, Ninth National Conference on Artificial Intelligence, Anaheim, CA, July 1991, pp. 67–73.

[Fischer, Nieper-Lemke 1989]
G. Fischer and H. Nieper-Lemke, *HELGON: Extending the Retrieval by Reformulation Paradigm*, Human Factors in Computing Systems, CHI'89 Conference Proceedings (Austin, TX), ACM, New York, May 1989, pp. 357-362.

[Fischer, Stevens 1987]
G. Fischer and C. Stevens, *Volunteering Information — Enhancing the Communication Capabilities of Knowledge-Based Systems*, Proceedings of INTERACT'87, 2nd IFIP Conference on Human-Computer Interaction, Stuttgart, FRG, September 1987, pp. 965-971.

[Fischer, Stevens 1991]
G. Fischer and C. Stevens, *Information Access in Complex, Poorly Structured Information Spaces*, Human Factors in Computing Systems, CHI'91 Conference Proceedings, ACM, New Orleans, LA, April 1991, pp. 63-70.

[Foltz 1990]
P.W. Foltz, *Using Latent Semantic Indexing for Information Filtering*, Proceedings of the Conference on Office Information Systems, ACM, Cambridge, MA, 1990,

[Fox 1989]
C. Fox, *A Stop List for General Text*, SIGIR Forum, Vol. 24, No. 1–2, Fall 1989, pp. 19–35.

[Furnas et al. 1987]
G.W. Furnas, T.K. Landauer, L.M. Gomez and S.T. Dumais, *The Vocabulary Problem in Human-System Communication*, Communications of the ACM, Vol. 30, No. 11, November 1987, pp. 964-971.

[Girgensohn 1992]

A. Girgensohn, *End-User Modifiability in Knowledge-Based Design Environments*, Thesis, Department of Computer Science, University of Colorado1992.

[Greene et al. 1990]
S.L. Greene, S.J. Devlin, P.E. Cannata and L.M. Gomez, *No IFs, ANDs, or ORs: A Study of Database Querying*, International Journal of Man-Machine Studies, Vol. 32, No. 3, 1990, pp. 303-326.

[Grudin 1989]
J. Grudin, *Why CSCW Applications Fail: Problems in the Design and Evaluation of Organizational Interfaces*, Office: Technology and People, Vol. 4, No. 3, 1989, pp. 245–264.

[Guindon 1988]
R. Guindon, *How To Interface To Advisory Systems? Users Request Help With A Very Simple Language.*, Human Factors in Computing Systems, CHI'88 Conference Proceedings, ACM, Washington, DC, May 1988, pp. 191–196.

[Halasz 1987]
F.G. Halasz, *Reflections on NoteCards: Seven Issues for the Next Generation of Hypermedia Systems*, Hypertext'87 Papers, University of North Carolina, Chapel Hill, NC, November 1987, pp. 345-365.

[Henninger 1990]
S. Henninger, *Defining the Roles of Humans and Computers in Cooperative Problem Solving Systems for Information Retrieval*, Proceedings of the AAAI Spring Symposium: Workshop on Knowledge-Based Human Computer Communication, March 1990,

[Hill, Miller 1988]
W.C. Hill and J.R. Miller, *Justified Advice: A Semi-Naturalistic Study of Advisory Strategies*, Human Factors in Computing Systems, CHI'88 Conference Proceedings, ACM, Washington, DC, May 1988, pp. 185–190.

[Horvitz 1989]
R. Horvitz, *The Usenet Underground*, Whole Earth Review, Vol. No. 65, Winter 1989, pp. 112–115.
[Iselin 1989]
E. Iselin, *The Impact of Information Diversity on Information Overload Effects In Unstructured Managerial Decision Making*, Journal of Information Science, Vol. 15, No. 1989, pp. 163–173.

[Kahneman, Tversky 1973]
D. Kahneman and A. Tversky, *On The Psychology Of Prediction*, Psychological Review, Vol. 80, No. 4, July 1973, pp. 237–251.

[Kantor, Lapsley 1986]

B. Kantor and P. Lapsley, *Network News Transfer Protocol: A Proposed Standard for the Sream–Based Transmission of News Messages*, Network Working Group Request For Comments 977, February 1986.

[Kaplan etal 1990]
S.J. Kaplan, M.D. Kapor, E.J. Belove, R.A. Landsman and T.R. Drake, *Agenda: A Personal Information Manager*, Communications Of The ACM, Vol. 33, No. 7, July 1990, pp. 105–116.

[Kass, Finin 1987]
R. Kass and T. Finin, *Modeling the User in Natural Language Systems*, Computational Linguistics, Special Issue on User Modeling, Vol. 14, No. 3, 1987, pp. 5-22.

[Kass, Stadnyk 1992]
R. Kass and I. Stadnyk, *Intelligent Assistance for the Communication of Information in Large Organizations*, Proceedings of the Eighth IEEE Conference on Artificial Intelligence for Applications, 1992, pp. 171–178.

[Keene 1989]
S.E. Keene, *Object–Oriented Programming In Common Lisp*, Addison–Wesley, 1989.

[Kelly 1987]
J.P. Kelly, *An Analysis of Organizational Factors Associated with the Use of Electronic Bulletin Boards in Educational Administration Environments*, Phd Thesis, The Graduate College, Texas A&M University, December 1987.

[Kilari 1989]
P.V. Kilari, *The Effects of Information Load on Decision–Makers and the Use of Heuristics*, Phd Dissertation Thesis, Management Science, The University of Wisconsin, 1989.

[Klapp 1986]
O.E. Klapp, *Overload and Boredom*, Greenwood Press, New York, 1986.

[Lemke, Fischer 1990]
A.C. Lemke and G. Fischer, *A Cooperative Problem Solving System for User Interface Design*, Proceedings of AAAI-90, Ninth National Conference of Artificial Intelligence, 1990,

[Mackay 1990a]
W.E. Mackay, *Patterns of Sharing Customizable Software*, Proceedings of the Conference on Computer-Supported Cooperative Work (CSCW'90), ACM, New York, October, 1990 1990a, pp. 209-221.

[Mackay 1990b]

W.E. Mackay, *Users and Customizable Software: A Co–adaptive Phenomenon*, Phd Thesis, Alfred P. Sloan School of Management, Massachusetts Institute of Technology, May 1990b.

[Malone, Grant, Turbak 1986]
T.W. Malone, K.R. Grant and F.A. Turbak, *The Information Lens: An Intelligent System for Information Sharing in Organizations*, Human Factors in Computing Systems, CHI'86 Conference Proceedings (Boston, MA), ACM, New York, April 1986, pp. 1-8.

[Mastaglio 1990]
T.W. Mastaglio, *A User Modelling Approach for Computer–Based Critiquing*, Ph.D. Thesis, Computer Science Department, University of Colorado at Boulder, 1990.

[Moran 1983]
T.P. Moran, *Getting into a System: External-Internal Task Mapping Analysis*, Human Factors in Computing Systems, CHI'83 Conference Proceedings (Boston, MA), New York, December 1983, pp. 45-49.

[Nakakoji, Fischer 1990]
K. Nakakoji and G. Fischer, *Catalog Explorer: Exploiting the Synergy of Integrated Design Environments*, Software Symposium'90 (Kyoto, Japan), June, 1990 1990, pp. 264-271.

[Nieper 1985]
H. Nieper, *TRISTAN: A Generic Display and Editing System for Hierarchical Structures*, Department of Computer Science, University of Colorado, 1985.

[Norman 1986]
D.A. Norman, *Cognitive Engineering*, in S.W.D. D.A. Norman (eds.), *User Centered System Design, New Perspectives on Human-Computer Interaction*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1986, Ch. Chapter 3.

[Norr 1992]
H. Norr, *Workplaces Beating Back Virus Threat*, MacWEEK, Vol. 6, No. 40, November 9 1992, pp. 4.
[Piekara, Strube 1987]
F.H. Piekara and G. Strube, *Data–Base Organization and Cognitive Structure: Using Information Systems Organized By Oneself and Others*, Proceedings of INTERACT'87, 2nd IFIP Conference on Human-Computer Interaction (Stuttgart, FRG), Amsterdam, September 1987, pp. 965-971.

[Quarterman, Hoskins 1986]
J.S. Quarterman and J.C. Hoskins, *Notable Computer Networks*, Communications of the ACM, Vol. 29, No. 10, October 1986, pp. 932–971.

[Reddy 1983]

R. Reddy, *Technologies for Learning*, Computers in Education: Realizing the Potential, Report of a Research Conference, Washington, D.C., August 1983, pp. 49-60.

[Schick, Gordon, Haka 1990]
A.G. Schick, L.A. Gordon and S. Haka, *Information Overload: A Temporal Approach*, Accounting Organizations and Society, Vol. 15, No. 3, 1990, pp. 199–220.

[Schroder etal. 1967]
H.M. Schroder, M.J. Driver and S. Streufert, *Human Information Processing*, Holt, Rinehart and Winston, New York, 1967.

[Schwartz 1989]
M.F. Schwartz, *The Networked Resource Discovery Project*, Proceedings of the IFIP XI World Congress, San Francisco, California, August 1989, pp. 827-832.

[Simon 1981]
H.A. Simon, *The Sciences of the Artificial*, The MIT Press, Cambridge, MA, 1981.

[Stallman 1981]
R.M. Stallman, *EMACS, the Extensible, Customizable, Self-Documenting Display Editor*, ACM SIGOA Newsletter, Vol. 2, No. 1/2, 1981, pp. 147-156.

[Steele 1990]
G.L. Steele, *Common Lisp The Language*, Digital Press, Bedford, Massachusetts, 1990.

[Streufert 1973]
S.C. Streufert, *Effects of Information Relevance on Decision Making in Complex Environments*, Memory and Cognition, Vol. 3, No. 1973, pp. 224–228.

[Thomas, Kellogg 1989]
J.C. Thomas and W.A. Kellogg, *Minimizing Ecological Gaps in Interface Design*, IEEE Software, Vol. 78, No. 1, January 1989, pp. 78–86.

[Thukral 1983]
V.K. Thukral, *Cognitive Strain as a Cause of Negative Bias*, Phd Dissertation Thesis, School of Business, University of Kansas, February 1983.

[Trigg, Moran, Halasz 1987]
R.H. Trigg, T.P. Moran and F.G. Halasz, *Adaptability and Tailorability in NoteCards*, Proceedings of INTERACT'87, 2nd IFIP Conference on Human-Computer Interaction (Stuttgart, FRG), Amsterdam, September, 1987 1987, pp. 723-728.

[Whitney 1978]
D.C. Whitney, *Information Overload in the Newsroom: Two Case Studies*, Phd Dissertation Thesis, The Graduate School, University of Minnesota, December 1978.

[Winograd, Flores 1986]

T. Winograd and F. Flores, *Understanding Computers and Cognition: A New Foundation for Design*, Ablex Publishing COrporation, Norwood, NJ, 1986.

[Wonnacott, Wonnacott 1977]
T.H. Wonnacott and R.J. Wonnacott, *Introdictory Statistics for Business and Economics*, John Wiley & Sons, New York, 1977.

[Wroblewski, McCandless, Hill 1989]
D. Wroblewski, T. McCandless and W. Hill, Agency Considered Harmful, AAAI Symposium on Knowledge–based Human Computer Communication Position Paper, 1989, Unpublished.

[Wyle, Frei 1989]
M.F. Wyle and H.P. Frei, *Retrieving Highly Dynamic, Widely Distributed Information*, Proceedings of the Twelfth Annual International ACM SIGIR Conference, Cambridge, Mass., June 25–28 1989, pp. 108–115.

13. Appendix 1: About Macintosh Usenet Groups

This section contains a message that is posted each month to the Usenet newsgroup comp.sys.mac.announce for the purpose of establishing the Usenet culture ground rules for the Macintosh newsgroups. Originally, this type of posting was made to the newsgroup news.announce.newusers but the Macintosh community decided when their newsgroups were split into many groups that they needed an announcement of their own. It is included here so that you may get some idea of how the Usenet culture develops.

Article: 27 of comp.sys.mac.announce
From: geoff@pmafire.inel.gov (Geoff Allen)
Newsgroups: comp.sys.mac.announce
Subject: About Macintosh Usenet Groups
Summary: General Introduction. Last 'real' change October 1.
Date: 1 Mar 91 08:33:11 GMT
Expires: 05 Apr 91 08:33:05 GMT
Followup-To: comp.sys.mac.misc
Lines: 253
Approved: by the moderator, posted by geoff@pmafire.uucp
Supersedes: <1991Feb04.161052.10103@pmafire.inel.gov>

[Made some minor editorial changes Nov. 1, 1990. Last real change in content was Oct. 1, 1990.]

This is a regular posting providing information about Usenet Macintosh groups. It is designed to help you find the best place to post your Macintosh questions/discussions. A companion posting answers some frequently asked questions.

Feel free to send comments, questions, etc. about this posting to geoff@pmafire.inel.gov (or uunet!pmafire!geoff).

Before going to the group descriptions, let's look at some general points.

If You're New to Usenet

If you're new to Usenet, be sure to read news.announce.newusers and news.newusers.questions. You'll find lots of valuable information there, along with some friendly folks who can help with the things we all had to learn.

Crossposting

You should have little need to crosspost within the Macintosh newsgroups. Try to find the group that most specifically fits your topic and post to it only. If you have a programming question, post it to comp.sys.mac.programmer. Don't crosspost to comp.sys.mac.misc ``just to be sure everyone sees it.'' There are folks reading comp.sys.mac.misc who don't really care about programming the Macintosh, and they

shouldn't have to wade through programming discussions. That's why comp.sys.mac.programmer exists.

This, of course, applies to the other groups as well.

For Sale Postings

For sale items should be posted to a .forsale group (e.g. misc.forsale, ba.forsale, or, especially, misc.forsale.computers). That's why those groups exist. Also, please limit the distribution of for sale postings. It probably isn't too practical for someone in New Zealand to buy the Mac II you're selling in California!

Usenet Macintosh groups

There are several newsgroups for discussions related to the Macintosh. Here are group names, along with their ``official'' definitions taken from the monthly ``List of Active Newsgroups'' posted in news.announce.newusers, news.lists, and news.groups. Each of these is followed by a more detailed explanation of the newsgroup.

****

comp.sys.mac Discussions about the Apple Macintosh & Lisa.

Don't post here. This group shouldn't even exist on your machine any more. It's history.

****

comp.sys.mac.misc General discussions about the Apple Macintosh.

This is the main Macintosh newsgroup. This is where general Macintosh discussions (i.e., those without a specific group) should go. Volume is very high in this group, so please try to do your part by not posting here if a more specific group exists.

****

comp.sys.mac.announce Important notices for Macintosh users. (Moderated)

The exact scope of what constitutes an ``important notice for Macintosh users'' has yet to be fully defined.

The moderator is Werner Uhrig. Your news software should automatically mail anything posted to this group to him. If it doesn't, you can mail your message to csma@rascal.ics.utexas.edu.

****

comp.sys.mac.apps Discussion of Macintosh applications.

If it's an application (and doesn't have another group in which it fits, such as HyperCard in comp.sys.mac.hypercard, programming tools in comp.sys.mac.programmer, and communications programs in comp.sys.mac.comm), this is the place to talk about it.

****

comp.sys.mac.comm Discussion of Macintosh communications.

This includes telecommunications and networking.

****

comp.sys.mac.digest Apple Macintosh: info&uses, but no programs. (Moderated)

This newsgroup (which is actually a gatewayed mailing list) presents a digest of postings covering all Macintosh topics. It is also the source of information for files available for FTP from sumex-aim.stanford.edu. (See the Frequently Asked Questions posting for information on FTP and alternatives for those not on the Internet.)

The moderators' address (in case your news software won't mail your postings for you) is info-mac@sumex-aim.stanford.edu.

****

comp.sys.mac.games Discussions of games on the Macintosh.

This one is pretty self-explanatory.

****

comp.sys.mac.hardware Macintosh hardware issues & discussions.

This newsgroup is for discussing Macintosh hardware. If you have a question about your ImageWriter or want to know about upgrading the memory in your Mac, this is the place.

****

comp.sys.mac.hypercard The Macintosh Hypercard: info & uses.

This one is pretty self-explanatory. If you have a question or comment about HyperCard, it belongs here. By extension, discussions about SuperCard and Plus (two HyperCard-like programs from other vendors) also belong here.

****

comp.sys.mac.programmer Discussion by people programming the Apple Macintosh.

This group is for those involved in Macintosh programming. (If your Macintosh programming is in HyperTalk, then you should post to comp.sys.mac.hypercard. Questions and discussions specifically related to writing XCMDs and XFCNs for HyperCard should also go in comp.sys.mac.hypercard.)

****

comp.sys.mac.system Discussions of Macintosh system software.

This group is for anything that goes in your system folder (INITs, CDEVs, DAs, etc.) in addition to Macintosh System software (System 6.0.x, System 7.0, Finder, Multifinder, etc.).

****

comp.sys.mac.wanted Postings of "I want XYZ for my Mac."

This is the place to post, ``I missed part x of ... on comp.binaries.mac", ``I need a ...", or anything else that fits the ``wanted" label.

****

comp.binaries.mac Encoded Macintosh programs in binary. (Moderated)

Macintosh programs in BinHex form appear here. For more information about this newsgroup, see the Frequently Asked Questions posting.

The moderator of this group is Roger Long. If your news software doesn't mail your message, you may send it to macintosh@felix.uucp.

****

comp.sources.mac Software for the Apple Macintosh. (Moderated)

This group is for text listings of Macintosh source code. It hardly ever sees any traffic. This is because people posting source code usually post StuffIt archives of the actual source files (often along with an executable version of the program for those without compilers). These would then go to comp.binaries.mac. Generally the only things you'll see here are UNIX sources of interest to Macintosh users.

The moderator of this group is Roger Long. If your news software doesn't mail your message, you may send it to macintosh@felix.uucp.

****

comp.protocols.appletalk Applebus hardware & software.

Questions and discussions relating to AppleTalk (LocalTalk) and Macintosh networking belong here.

****

comp.unix.aux The version of UNIX for Apple Macintosh II computers.

If you want to talk about A/UX, this is the place.

****

Other Groups of Interest

There are other groups that, while not Macintosh-specific, may be of interest. For example:

comp.fonts Typefonts -- design, conversion, use, etc.
comp.text Text processing issues and methods.
comp.text.desktop Technology & techniques of desktop publishing.

These groups would be a better place for desktop publishing discussions, since they exist for just such a purpose.

comp.lang.c Discussion about C.
comp.lang.pascal Discussion about Pascal.
(or .fortran, .c++, .smalltalk, ... You get the idea.)

If you have questions or discussions about the language you're programming in rather than a Macintosh-specific programming question, the appropriate comp.lang.* group would be a better place to post.

comp.protocols.* groups might be of value of you're doing much communicating between systems.

In short, check the list of active newsgroups (posted monthly to news.announce.newusers, news.lists, and news.groups) for newsgroups that might be of interest. Some other examples you might wish to explore include:

comp.graphics	Computer graphics, art, animation, image processing.
comp.graphics.digest	Graphics software, hardware, theory, etc. (Moderated)
comp.ivideodisc	Interactive videodiscs -- uses, potential, etc.
comp.lang.postscript	The PostScript Page Description Language.
comp.laser-printers	Laser printers, hardware & software. (Moderated)
comp.object	Object-oriented programming and languages.
comp.os.mach	The MACH OS from CMU & other places.
comp.os.minix	Discussion of Tanenbaum's MINIX system.

[These two represent UNIX-like systems available for the Macintosh.]

comp.unix.* groups

To learn more about the workings of A/UX, or to learn more about the operating system most of us use when we read news. Especially check out comp.unix.questions if you're new to UNIX.

comp.windows.misc	Various issues about windowing systems.
comp.windows.x	Discussion about the X Window System.
comp.virus	Computer viruses & security. (Moderated)

--
Geoff Allen
geoff@pmafire.inel.gov

14. Appendix 2: Answers to Frequently Asked Questions

This section contains a message that is posted each month to the Usenet newsgroup comp.sys.mac.announce for the purpose of keeping frequently asked questions from popping up several times a month. This was deemed necessary after years of repeating many of these questions/answers. It is included here so that you may get some idea of how the Usenet culture develops.

Article: 28 of comp.sys.mac.announce
From: geoff@pmafire.inel.gov (Geoff Allen)
Newsgroups: comp.sys.mac.announce
Subject: Answers to Frequently Asked Questions
Summary: Last change, Feb. 28, 1991
Message-ID: <1991Mar01.083519.8193@pmafire.inel.gov>
Date: 1 Mar 91 08:35:19 GMT
Expires: 05 Apr 91 08:35:08 GMT
Followup-To: comp.sys.mac.misc
Organization: WINCO
Lines: 429
Approved: by the moderator, posted by geoff@pmafire.inel.gov
Supersedes: <1991Feb04.161143.10193@pmafire.inel.gov>

[Lots of changes this month. It's probably worth reading this, even if you're familiar with past versions.

Changes I've made are:

A very major change in the question about Folders From Hell.

Two new questions. One about the ftp site um-mts.cc.umich.edu, and the other about converting word processing file formats to TeX or troff and back.

Lots of minor changes to things throughout the file.

Enjoy!

-- Geoff]

This document answers some frequently asked questions from the comp.sys.mac newsgroups. These answers generally haven't originated with me. I've primarily served as an editor, putting together my own knowledge and some collective net wisdom. If your favorite FAQ isn't here, write it up (along with an answer) and send it to me for possible inclusion.

The questions answered in this document are:

Q: How do I print PostScript to a file instead of a laser printer?

Q: I have a folder I can't delete. What should I do?

Q: I took my document to another Mac so I could print it on a

LaserWriter, and the formatting was all messed up. What did I do wrong?

Q: What do I do with the files on comp.binaries.mac (or *.hqx files that I FTP'd)?

Q: Where can I get BinHex 4.0?

Q: What does ".hqx" (or ".sit", etc.) mean?

Q: What is FTP?

Q: What are some sites from which I can FTP Macintosh software?

Q: Where's the System software on apple.com?

Q: How do I get stuff from um-mts.cc.umich.edu?

Q: I'm at a .UUCP site; can I use FTP?

Q: How do I post to comp.binaries.mac?

Q: What is the proper format for submissions to comp.binaries.mac?

Q: Why is my posting to comp.binaries.mac taking so long to show up? Did it get lost in the mail?

Q: How can I convert a file from my word processor format to TeX (or troff)? (Or the reverse.)

~~~~~~~~~~

Q: How do I print PostScript to a file instead of a laser printer?

A: Make sure that the LaserWriter is chosen in the Chooser DA (even if you don't have a LaserWriter). Immediately after clicking OK in the Print dialog box, press and hold the `f' key (`k' if you want the LaserPrep header information included in the document). You should see a dialog box telling you that a postscript file is being created. The file will be named ``Postscript0" and you may need to use the Find File DA (or equivalent) to locate it. You can then send it to a LaserWriter on a Mac using a program called SendPS. For use on Unix systems with other PostScript printers, check out the program macps, available from sumex and other places.

If you're running MultiFinder, this will only work if you have background printing turned off.

For a lot more information about generating PostScript on the Mac, check out the file /info-mac/tips/generating-postscript.txt from sumex-aim.stanford.edu. (If you don't understand what I just said, see the question on FTP below.) This file has a lot of useful information and is definitely worth your time and effort to get.

Q: I have a folder I can't delete. What should I do?

A: This is the infamous ``Folder From Hell" problem you may see mentioned from time to time. The problem is usually that the Finder's count of the number of files in the folder gets messed up (including being negative). The Finder will only trash folders that it believes contain 0 folders. There are about 5,000,000 suggested ways to get rid of Folders From Hell. I used to include several of them in this post. Now I offer one (it's *that* good):
~~~~~~~~~~

First, make sure that the folder is empty! It may contain hidden files in it. (Hidden files can't be seen on the Desktop, but can be found with various utilities, including ResEdit, MacTools, DiskTop, etc.)

Create a new folder on another drive or in another folder with the same name as the Folder From Hell. Drag this new folder to the same folder/disk that contains the Folder From Hell. The Mac will ask you if you want to replace items with the same name. Of course you do. Voila! No more Folder From Hell.

I wish I had thought of it! (Special accolades to Matt Howard <HOWIE@triton.tamu.edu> for suggesting this first.)

If you're still interested in the other techniques, they include holding down command and option while dragging it to the trash (useful in other situations too), putting a few files into the folder and then trashing them (which may reset the file count to 0 if it was negative), trashing the folder after booting off another disk, stuffing the folder with StuffIt, and resetting the folder's file count with ResEdit or MacSnoop (only for the stout of heart :^)).

There is also a program called HellFolderFix, which claims to correct the problem.

The final ``if all else fails'' move is to back up the disk, erase it, and restore it.

These latter techniques are here for the curious. I don't see any need for them any more. The replacement folder trick should work.

Q: I took my document to another Mac so I could print it on a LaserWriter, and the formatting was all messed up. What did I do wrong?

A: The Macintosh uses slightly different formatting for LaserWriters and ImageWriters. If you are going to print the file on a LaserWriter, choose the LaserWriter in the Chooser DA (whether you have a LaserWriter or not) when composing your document. Then the formatting will be correct for the LaserWriter.

Selecting ``Tall Adjusted'' in the ImageWriter Page Setup dialog seems to work also. Can anyone positively confirm or deny this?

Q: What do I do with the files on comp.binaries.mac (or *.hqx files that I FTP'd)?

Q: Where can I get BinHex 4.0?

A: These two questions are related, so they will be answered together.

I know that the file you've got says,

(This file must be converted with BinHex 4.0)

but the truth is that you most likely don't need BinHex 4.0. What you really need is StuffIt or UnStuffit (or the Deluxe or Classic versions of either). Here's why:

StuffIt will encode and decode files into BinHex format. The large majority of files you'll see are Stuffed, in addition to being BinHexed, so you will need StuffIt (or the free UnStuffIt) anyway.

(The differences between the programs are:

StuffIt is the original shareware compression program which has become the standard.

UnStuffIt is a free program which only unstuffs files.

StuffIt Deluxe is the commercial version of StuffIt.

StuffIt Classic is the latest shareware version of StuffIt.

There's also an UnStuffIt Deluxe, which is analogous to UnStuffIt. There may also be an UnStuffIt Classic.

The ``Un'' versions of StuffIt *don't* convert BinHex files, so for translating net stuff, I'd recommend getting StuffIt or you can get one of the BinHex conversion programs mentioned in the next question.

Adding to the confusion caused by all the versions of StuffIt out there is the existence of new compression programs which are starting to gain popularity. One that you'll probably hear about is Compact Pro (formerly Compactor), which is a shareware contender to the StuffIt throne. See the next question for what some common filename extensions mean.

Anyway, back to the original topic... :^)

Once you get the files to your Macintosh, join all the pieces into one file using your word processor (be sure to edit out everything that doesn't look like gibberish). The first and last character of all the gibberish stuff (the actual BinHex encoding) should be a colon (":"). Save this file in ``text only'' format. Then run one of the StuffIt programs and use the ``Decode BinHex File'' option on the ``Other'' menu. This should give you a StuffIt file (or regular file, if the original was not Stuffed). You can then unstuff the file using (Un)Stuffit.

Compact Pro can also decode BinHex. You have to delete everything before ``(This file...'' info to use it, though (you don't need to with StuffIt).

Another alternative, for those using Unix, is to get the program mcvert (available as a shar file from sumex and other places). Mcvert will convert your BinHex files to MacBinary before you download them. It also saves you the trouble of joining the files (you simply specify them as command arguments -- no editing is necessary), and is much faster than converting the files on the Mac. The MacBinary file will also be smaller, which saves transmission time to your Mac.

Q: What does ".hqx" (or ".sit", etc.) mean?

A: Here are some common file suffixes that you are likely to see and what they mean:

.hqx	BinHex format -- use BinHex 4.0, BinHex DA, any of the StuffIt programs, Compact Pro, or mcvert (for Unix).
.sit	StuffIt format -- use any of the StuffIt programs or unsit (for Unix). (There is also a program which will convert StuffIt to Compact Pro format, if you're using Compact Pro.)
.cpt	Compactor format -- use Compact Pro or Extractor
.sea	Compact Pro-generated self extracting archive. Double-clicking on this should give you the

uncompressed file.

.pit PackIt format, not used much any more -- use PackIt or any of the StuffIt programs.

.image DiskCopy images of a floppy disk. This is the format used for the System software on apple.com. Use the DiskCopy application to translate these back onto disks.

Q: What is FTP?

A: FTP stands for File Transfer Protocol. It is only available if your site is on the Internet (check with your site administrator if you aren't sure). Usually you'll hear that a site has items available for ``anonymous ftp.'' This means that to get something from that site, you:

1. type 'ftp [site-name]' (e.g. 'ftp sumex-aim.stanford.edu')

   [If this doesn't work, it may be that you need to type the Internet address of the site, rather than its name. For example, sumex-aim.stanford.edu is 36.44.0.6.]

2. Log in as 'anonymous' and use anything for a password (convention says to use your login id, e.g. joe@blow.com).

3. go to the specified directory (using cd, etc.)

4. type 'get [filemane]' (e.g. get disinfectant-24.hqx)

You should then have the file. Check your local man page or guru for more on ftp at your site, since specific implementations may vary somewhat.

Also, please, only ftp during non-business hours. Ftp is a privilege granted to you by these sites. Don't beat up on them during working hours.

Q: What are some sites from which I can FTP Macintosh software?

A: The main sites are:

sumex-aim.stanford.edu [36.44.0.6], which contains a large collection of public domain and shareware programs.

ftp.apple.com, (coming soon -- until then look for stuff on apple.com [130.43.2.2]) which contains many items provided by Apple (System Software, Technical Notes, etc.),

um-mts.cc.umich.edu [35.1.1.43], which has more stuff than you could ever imagine. It's a different beast, though; see below.

and

wsmr-simtel20.army.mil [26.2.0.74]

Items at sumex and apple are in BinHex form. Items at simtel20 are in binary form; be sure to execute the 'binary' or 'image' command from within the ftp program before getting them.

Q: Where's the System software on apple.com?

A: It's in the directory /pub/dts/sw.license. Not the most intuitive place, I know. :^)

(By the way, you should download and read the license agreement before you get the System software. Also be sure you have a copy of the DiskCopy application found in this directory, so that you can translate the .image files back into disks.)

Q: How do I get stuff from um-mts.cc.umich.edu?

A: Like I said, it's a different beast. It fools you because everything is in one *huge* directory, even though there are slashes ('/') in the file names. Here's what to do there (based on ftp-ing from a Unix system. I have no idea what it's like from VMS):

```
cd pc2:
        [This puts you where you can get the Mac stuff.]
get !index umich-index
        [Transfer the index. Believe me, you don't want to do
        an 'ls' here! The 'umich-index' tells Unix what to name
        it on your machine. If you leave it out, you'll get a
        file named '!INDEX', which will cause you lots of grief
        if you use the C-shell.]
bye
```

Now peruse the index. You'll notice that files have names like FO/NIFTY-FONT. That slash in the filename will freak Unix out. If you simply type 'get fo/nifty-font', you'll get an error message stating that the file couldn't be created (unless you happen to have a directory named 'fo' in your current directory). To deal with this, just use the above suggestion, type 'get fo/nifty-font nifty-font.hqx' (or whatever filename you want to give it).

The files are all in BinHex form.

Q: I'm at a .UUCP site; can I use FTP?

A: No. But there are alternatives. Perhaps the most useful is the listserver at Rice. It mirrors the archive at sumex, and is updated every night. You can reach it at:

```
LISTSERV@ricevm1.rice.edu (or ricevm1.rice.edu!LISTSERV)
```

The message you send should be of the form:

```
$MACARCH GET [what-you-want-to-get]
```

For example:

```
$MACARCH GET $MACARCH.CONTENTS
```

will give you a listing of the contents of the Mac archive.

```
$MACARCH GET VIRUS/DISINFECTANT-24.HQX
```

will give you a binhex'ed copy of Disinfectant 2.4

```
$MACARCH HELP
```

will give you a help message that may be more confusing than helpful (at least *I* found it confusing). But I was contacted by the Mac archive's maintainer, and he's looking to make the help message more helpful. Part of the problem is that it's intended more for direct Bitnet users rather than mail users. That's one of the main things that confused me. Anyway, he plans to have two help messages, one long one and one short one. I'll update the info here when that happens.

There is a limit of 256 Kb/person/day.

The listserver appears to be case insensitive, but the help info and the listings in MACARCH.CONTENTS give the commands as all uppercase, so I just use uppercase.

If the file has lines that are over 80 characters long, LISTSERV will put the file into ``Listserv Punch'' format. To decode this on a Mac, there was a program on sumex to handle this, but it doesn't seem to be there any more. If you're using a Unix machine, send me e-mail and I'll send you an awk script that I wrote to handle the conversion.

Another alternative for folks without FTP access is to use the FTP server at princeton. Send a message to bitftp@pucc.princeton.edu (or bitftp@pucc if you're mailing from Bitnet). The first message you send should be the word ``HELP'' (all uppercase), on the first line starting in the first column. Read that, and then (hopefully) you'll be ready to send ftp requests to their server. I personally haven't tried it, but from what I've seen, it looks handy.

Q: How do I post to comp.binaries.mac?

A: Your news software should handle this for you. Posting to the group should automatically get your message mailed to the moderator.

If this does not happen on your system, you can mail your posting to the moderator yourself. The moderator of comp.binaries.mac is Roger Long and his address (for submissions only) is

macintosh@felix.uucp

Q: What is the proper format for submissions to comp.binaries.mac?

A: Submissions should be in BinHex form. If the file is long, it may need to be split into two or more parts to get through some mail gateways (<100K per part should work). If you do split your file, put a line at the end of each part like ``End of part 1'' and a line at the beginning of each part like ``Beginning of part 2.'' This will help the moderator to know that everything made it through the mail, and to put it all back together again.

Use the original StuffIt format for compressing files before sending them. Not everyone has the newer programs, and StuffIt is still the lowest common denominator.

Q: Why is my posting to comp.binaries.mac taking so long to show up? Did it get lost in the mail?

A: Probably not. The group has a sizable backlog of items to post and, according to the moderator, is limited (administratively) to a volume of about 2.5Mb/month (which works out to about 84K/day). This means that only so much can be sent out each day. So your posting is probably not lost; it's just waiting its turn.

Q: How can I convert a file from my word processor format to TeX (or troff)? (Or the reverse.)

A: This is a FAQ in the Mac groups *and* in comp.text.tex. It's most common incarnation deals with RTF <=> TeX, since Microsoft's Rich Text Format resembles TeX.

Given all the interest, there really isnt much available. A recent posting by bin@primate.wisc.edu (Brain in Neutral) tells of Unix

programs written by dubois@primate.wisc.edu (Paul Dubois) available on indri.promate.wisc.edu [128.104.230.11] in /pub/RTF. These programs take RTF input and output nothing; plain text; troff; and character, word, and paragraph count. They are described as not yet being polished, and comments are welcomed.

If you want to be a net.hero, just write a translator between RTF and TeX. The opportunity awaits! :^)

--
Geoff Allen
uunet!pmafire!geoff
geoff@pmafire.inel.gov

15. Appendix 3: Conference Announcement Classification Experiment

From stevens Wed Nov 28 15:52:39 1990
Subject: Please spend a couple of minutes on this for me...
To: hcc
Date: Wed, 28 Nov 90 15:52:39 MST
X-Mailer: ELM [version 2.3 PL0]

As a small experiment I would like each of you to read the following conference announcement and answer two questions about it:

(1) If you were to post this announcement to a single newsgroup, what would that group be?

(2) If you really wanted to make sure the appropriate people received this announcement, and you were to do so by cross-posting to more than one newsgroup, which other newsgroups would you choose?

You can find a list of active newsgroups by (a) looking in your .newsrc file in your home directory (assuming that you add new newsgroups to the file when the system asks you what to do with new newsgroups) or (b) in the file newsigi:~stevens/tmp/active.

Don't spend too much time on this (5 minutes max). AND PLEASE DON'T TALK TO ANYONE ELSE ABOUT YOUR CHOICE (this means that you shouldn't send your response to the whole mailing list, just to me personally). For this experiment to work I need all of you to participate (there aren't really that many of us so please pitch in :->). Thanks very much in advance, I appreciate your time and effort.

========
| Curt |
========
==========

Workshop Announcement: Human-Computer Interface Design: Success Cases, Emerging Methods, and Real-World Context

This workshop will seek to answer the question, "How can we improve human-computer interface design methods to bring about more successful design?" We will begin by examining the design techniques employed for successful interfaces of the past, (e.g., Star). We then will look at a variety of emerging interface design methods, such as the use of rapid prototyping and analytical models. Finally, we will consider the pragmatic question of how successful design methods can be integrated into complex design organizations. In this final section, the space exploration program will be used as an example of large organizations designing interfaces to complex computer systems.

The workshop is being organized by the Institute of Cognitive Science at the University of Colorado, Boulder, and the Human-Computer Interaction Laboratory at the NASA Johnson Space Center. It will be held in Boulder, Colorado on July 24 - 26, 1991. The workshop will be attended by three groups -- interface designers; managers involved at several levels of interface design, development, and delivery; and HCI researchers. The workshop format will involve presentations of invited papers, as well as structured discussion of how existing and new interface design methods can meet the constraints of design organizations. The proceedings will be published in an edited book.

Specific Results

Subject	First Choice	Additional Choices

1	comp.infosystems	comp.groupware news.announce.conferences
2	comp.cog-eng	comp.windows.misc comp.ai cu.ics
3	cu.ics	boulder.hcc alt.hypertext cu.cs.grads sci.space
4	news.announce.conferences	comp.org.ieee
5	comp.ai	comp.ai.vision comp.lang.visual comp.theory.info-retrieval cu.asm cu.cs.clim cu.cs.general cu.cs.grads cu.cs.ugrads cu.ece.grads cu.general cu.ics cu.motif-talk cu.slug news.misc
6	news.announce.conferences	boulder.hcc co.general comp.ai cu.acm cu.cs.grads cu.ics
7	comp.cog-eng	sci.psychology.digest comp.graphics alt.graphics
8	news.announce.conferences	comp.groupware alt.hypertext comp.ai cu.ics
9	comp.ai.digest	cu.cs.general cu.ics
10	comp.ai	news.announce.conferences cu.ics co.general comp.lang.visual comp.windows.misc