Collaborative Classification and Evaluation of Usenet

Robert S. Brewer and Philip M. Johnson

Collaborative Software Development Laboratory
Department of Information and Computer Sciences
University of Hawaii
Honolulu, HI 96822
Tel: (808) 956-3489
Email: rbrewer@uhics.ics.hawaii.edu, johnson@hawaii.edu

ABSTRACT

Usenet is an example of the potential and problems of the nascent National Information Infrastructure. While Usenet makes an enormous amount of useful information available to its users, the daily data overwhelms any user who tries to read more than a fraction of it. This paper presents a collaboration-oriented approach to information classification and evaluation for very large, dynamic database structures such as Usenet. Our approach is implemented in a system called URN, a multi-user, collaborative, hypertextual Usenet reader. We show that this collaborative method, coupled with an adaptive interface, radically improves the overall relevance level of information presented to a user.

KEYWORDS: collaborative filtering, classification, evaluation, usenet, adaptive interfaces, large databases

INTRODUCTION

Information is rapidly becoming the essential currency of the modern world. With the growth of global connectivity, there is a growing desire to not only receive information, but to distribute one's own information. In the United States, there are proposals for a "National Information Infrastructure" [5] to enable its citizens to participate in the information age. However, there already exists a tremendously successful network allowing millions of people to send and receive information on a global scale. This network is Usenet.

Usenet is a global collaborative system *par excellence*. Usenet raises few barriers to participation beyond access to its technological infrastructure. Low-cost or free access services for home users, as well as connections to schools and libraries are increasing. Usenet does not require computer literacy beyond basic word processing skills, and does not restrict the content or dissemination of information. Anyone can make a posting about any topic, and anyone can read what anyone else has to say about a topic. Finally, anyone can create a new topic for discussion.

As a simple, imperfect analogy, imagine the daily papers from every major English-speaking city in the world arriving instantaneously on one's desktop each morning. (English is the predominant, but not exclusive language of Usenet.) The stack of papers consists of thousands of pages of text, photographs, and advertisements; local news stories from far away places; descriptions of and commentary on the same social, political, and/or technological event from hundreds of different perspectives; thousands of letters to editors from citizens, each expressing a different point of view on an issue of concern to them; questions, answers, rebuttals, and so on. Such information access would have a double-edged appeal: beyond access to new, helpful information on subjects for which one already has an interest, it would stimulate *new* interests by access to new subject areas, new events, new technology, and new cultural perspectives. It would be a potent force of basic human education and enrichment.

On the other hand, it would also be overwhelming. The sheer volume of news arriving every day would prevent even a cursory skimming in its entirety. Searching for information would be extremely time-consuming and frequently futile. Collecting information about a specific topic would necessitate highly heuristic, failure-prone strategies, such as "Read the London Times every day for an overview, and use these articles as pointers to regional papers of potential interest." Significantly, this problem of effective information access and retrieval is not a result of disorganization: newspapers are highly structured entities with both individual, local structure (summarized in its table of contents) and a common, global structure (most newspapers provide a "sports" section, a "classified advertisements" section, etc.) However, the structure of a newspaper, while well-suited to the needs of its immediate constituency, does not successfully scale up to the needs of the global community.

Current users of Usenet face the on-line equivalent of both the potential and problems of this hypothetical avid newspaper collector. Each day, thousands of new pages of text divided among thousands of topic areas (called "newsgroups") are generated and distributed to thousands of sites servicing millions of users. Each newsgroup is similar to a single newspaper with its own local structure and constituency. Information access and retrieval are similarly problematic, even though textual search mechanisms exist. This flood of information causes the problem we call *information overload*: too

much information presented in an unsuitable manner.

This paper presents findings from our research into effective utilization of the tremendous wealth of information in Usenet that is available theoretically, yet inaccessible practically. Our research approach recognizes that Usenet is an information system with properties very different from those of conventional database systems. Therefore it requires very different approaches to traditional database issues of information retrieval, information filtering, and information archiving. Our research thesis is that effective utilization of Usenet can be improved through explicitly collaborative efforts among small groups of people with similar interests who work together to retrieve, filter, and ultimately restructure information produced by Usenet into a form amenable to their own needs.

To pursue these research directions we created URN, a collaborative Usenet interface whose implementation and evaluation provides insights into this thesis. URN is designed to explore the representations and processes needed to provide a model of the interests of individuals within the group that can be used to predict the relevancy of future Usenet contributions. URN users *collaboratively* and incrementally create a shared, global representation of the content of each Usenet posting, but *individually* assess its relevancy to their own personal interests. The collaboration minimizes the overhead to any individual of this annotation, while improving the quality of the data used for relevancy assessment. Significantly, URN does not require users to agree upon a common single measure of relevancy—URN maintains a separate model of each user's interests. Instead, URN users focus their collaborative efforts on building a shared representation of each article's content and structure.

Analysis of data collected during an experimental trial of URN provides support for the validity of this approach. Over a two week period, URN incrementally built models of its six users' interests that provided increasingly accurate predictions of the relevancy of new articles to each user. The data also revealed many new insights into the issues surrounding effective information representation, retrieval, and filtering in Usenet. These insights provide useful new knowledge for designers of future Usenet readers, as well as for designers of future collaborative information management systems.

The next section of this paper provides background on Usenet, and discusses the differences between Usenet and traditional database systems that motivate our approach. The following section presents URN, a system implementing a novel, collaborative approach to effective utilization of Usenet. The following section describes the experimental evaluation of URN and its findings. The paper concludes with a discussion of the future directions for the URN project.

USENET AS AN INFORMATION SYSTEM

Background on Usenet

Usenet (standing for Users' Network) is a massive but loosely connected network of computers that exchange 'netnews' which can be thought of as a kind of 'public' email. Any user on a Usenet node can post an article to Usenet by simply typing in some text and submitting it to a program on the local computer. This local computer then forwards the article to a few close-by Usenet nodes, who in turn forward it to other nodes. In this manner news is propagated around the world, yet the original posting machine need only send it to a few near-by machines.

Although Usenet started in 1979 with only a few nodes, its growth has been incredible. As of March 1993, an estimated 76,000 Usenet sites existed with a total of over 2.4 million Usenet users [10]. In the two week period from January 10, 1994 to January 24, 1994, users generated over a gigabyte of data, consisting of approximately 673,000 separate articles [2].

Syntactic Structure of Usenet

Usenet articles are categorized into thousands of 'newsgroups' (almost 9000 newsgroups exist as of January 24, 1994 [2]). Newsgroup membership is the primary way to classify articles by subject area. Newsgroups are hierarchically named where '.' separates the levels of hierarchy. For example, the newsgroup about Macintosh hardware is called 'comp.sys.mac.hardware'. The subject areas of Usenet newsgroups is diverse, ranging from groups about software engineering (comp.software-eng) to groups about dogs (rec.pets.dogs) to groups about abortion (talk.abortion).

All articles on Usenet are ASCII text, but not all articles posted to Usenet are human-readable text; some are encoded versions of binary files: applications, pictures, and sounds. Regardless, all articles contain a 'subject line', which is intended to summarize the content of the article, other header information (such as the e-mail address of the posters, and the date of posting), and finally the 'body' of the posting containing the actual content.

The human-readable articles, as in all communications media, can be statements, questions, comments, replies to questions, poems, or any other textual object. After an article is posted, other users may choose to 'followup' that article with an article containing a reference to the original article. This process is recursive, with followups often generating new followups. A set of articles linked together in this way on a common topic is called a 'newsthread' or simply 'thread'. Sophisticated newsreading software (such as trn or GNUS) allows the user to navigate through Usenet newsgroups by following these threads.

Followup articles often include quotes, or partial copies of text from the original article. This quoting is usually done in an automated fashion so that readers can distinguish between quoted and original text. However, this creates a significant amount of redundant text, especially in long threads. In March 1993, for example, quotes represented more than 9% of Usenet's volume.

Semantic Structure of Usenet

Usenet is used for many different purposes, but three of the most common uses are question and answer, discussion, and dissemination. Many newsgroups consist partially or primarily of articles that ask questions about the topic, leading to followup articles by other users containing answers. This question and answer format can lead to problems, such as new

users posting a question that has been posted and answered previously. One technique employed to reduce this problem is a set of Frequently Asked Questions (FAQ). A list of FAQs with standard answers is posted to the newsgroup on a regular basis to ward off such questions.

Another use of Usenet is for discussion. These discussions are often quite long and involved, and can last for weeks or months. Due to their length and the number of users who participate, the topic of a thread can wander and evolve. In many cases there can be several topics under discussion simultaneously in the same thread. It would not be unusual for a thread discussing a software package to evolve into a debate on software patent law or even into a debate on the location of the best pizza parlor in Silicon Valley. Since newsreaders simply copy the subject line when creating a followup posting, the content of an article in a discussion thread frequently evolves quite far from that indicated by its subject line.

A final use of Usenet is for dissemination of timely information. For example, Usenet was used to disseminate information during such world events as the collapse of the Soviet Union, the initial reports on the 'discovery' of cold fusion, and the recent Los Angeles earthquake. These global events precipitate a deluge of articles on the subject, and usually a new newsgroup for the subject is created immediately.

Usenet vs. Conventional Databases

Many explanations for the problems in effectively utilizing the wealth of information generated by Usenet focus upon its immense volume and numbers of users, its global constituency, or even the mixture of information postings of transient interest (for example, an advertisement of an upcoming conference) with postings of more permanent interest (for example, a comparison of two programming paradigms.) However, such features alone do not explain why conventional database techniques cannot be applied to Usenet with equal success.

To illustrate this, note that airline reservation systems rival Usenet in volume of traffic, global constituency, number of users, and in mixing transiently useful information (single reservations) with more permanently useful information (airline routes and schedules). However, users of airline reservation systems do not share the problems faced by users of Usenet with respect to information access. Despite their immense size and complexity, airline reservation system users can almost always efficiently find all useful information in the system related to a particular topic.

The essential difference between airline reservation systems and Usenet, (and, indeed, between database systems in general and Usenet) is a fundamentally different notion of *structure*. The structure of an airline reservation system consists of a fixed set of schemas that represent all of the information contained in the system in a fine-grained, machine-processable form. This structure was decided upon before any information entered the system, and is fixed for the life of the system. The structure of airline reservation systems was designed after close analysis of this domain to ensure that all important forms of information access would be readily available. This combination of features enables airline reservation systems to scale up in size, complexity, and number of users without creating information overload or other problems manifested in Usenet.

In Usenet, however, both "structure" and "domain" have qualitatively more dynamic, emergent, and coarse-grained meanings. The structure of Usenet in database terms is its syntactic structure as described above: a set of newsgroups, each broken down into a set of articles, where each article is broken down into a set of generic, domain-independent fields. This structure is imposed solely to enable information *transport* across networks, hosts, and newsreader systems. It was not designed to support information retrieval, filtering, or analysis. In conventional database design, supporting information retrieval, filtering, or analysis is based upon careful domain analysis, followed by definition of specialized schemas and operations for that domain. However, the structure of Usenet must support not only the current 9000 domains, but also the thousands upon thousands of others to be created in the future.

If the techniques of conventional database systems for effective information management cannot apply to Usenet, what techniques do apply? The Usenet community has developed several ad-hoc solutions, as discussed next.

Current Usenet Information Management

The users of Usenet are well aware of their information overload problem, and a variety of information management mechanisms have evolved in response. Some examples are FAQ files, subscription, kill files, and quotation ratios.

As mentioned above, many groups address the problem of repetitious postings through the creation and periodic posting of FAQ files. FAQs probably do not substantially reduce the total volume of Usenet (although they do reduce the frequency of some of the most irritating postings for long-term users). Rather, they provide a means for novice users to quickly acquire some of the permanent, relatively slowly changing forms of information discussed in the group. FAQs never store information that becomes outdated quickly, nor do they store important or useful information that is not of general relevance. Finally, FAQs are typically maintained by a single person, and their quality, content, and currency is strictly dependent upon the commitment of that person.

A second mechanism for information management is newsgroup subscription. Since each newsgroup has a subject area, one can merely subscribe only to those newsgroups that one finds interesting, and leave the uninteresting groups unsubscribed. This technique works well only when the user has a small number of interests which are only discussed within a small number of newsgroups with a small daily volume. Most users have interests that potentially span many newsgroups, but subscribe only to those groups which most directly address issues of interest. As a simple example, a user interested in software engineering might read the newsgroup comp.software-eng. However, software engineering issues crop up frequently in hundreds of other newsgroups, including the comp.lang.* newsgroups, the news.software.* newsgroups, the comp.soft-sys.* newsgroups, and so forth. Subscription reduces the apparent volume of Usenet by simply

reducing the wealth of information available.

A third mechanism is "kill files". A kill file is a list of patterns that are designed to match fields in the header of an article such as its subject or author. Newsreaders do not present articles to users whose headers match one of the patterns in the kill file. In this way subjects or authors that a user find uninteresting are removed from his or her view.

There are three significant problems with kill files:

- *Kill files make all-or-nothing decisions.* The patterns for kill files must be chosen very carefully, or the user risks killing articles that are relevant to their interests. For example, a hypothetical user might dislike IBM, and therefore create a kill file entry that kills all articles related to the subject of IBM. However, this hypothetical user might also love Apple Computer. In this case, if an article is posted about a partnership between Apple and IBM the kill file would prevent the user from seeing it.
- *Kill files only deal with uninteresting articles.* The second problem with kill files is that they can only exclude articles; they cannot bring articles to the users' attention. Kill files only provide a structure for users to list patterns that exclude everything that they do not like, when it is easier for users to think about patterns that they do like.
- *Kill files are brittle.* Finally, kill files are difficult to create and maintain [11]. It is difficult, for example, to know when to take a inappropriate pattern *off* the list, since the user will not see the interesting articles deleted by the kill file.

A final method currently used for information management is the quotation-ratio restriction present in some newsreaders. This mechanism is designed to reduce the number of "me-too" comments by preventing articles from being posted unless the number of new lines of commentary in the article is greater than the number of quoted lines. This mechanism has completely backfired, since users quickly learn to simply "pad" their articles with lines of junk characters in order to trick the newsreader—thus exacerbating the very problem that the mechanism was intended to resolve.

COLLABORATIVE USENET PARTICIPATION WITH URN

To better understand the information overload problem and to explore approaches to effective information management in Usenet, we designed, implemented, and evaluated a collaborative system called URN. This section presents URN's requirements, design, implementation, and relationship to other Usenet interfaces. The next section will discuss our results from its evaluation.

Requirements

The design of URN is influenced by the following essential requirements, which we believe must be satisfied by any system providing effective information management in Usenet.

- *URN must provide a representation of the user.* The range of domains in Usenet, and the variety of users require an effective information management technique to explicitly represent users and their interests. Given the dynamic nature of both Usenet domains and user interests, the representation should be adaptive and provide confidence levels. For example, it should distinguish between topics for which the user evidences a strong interest, topics for which the user evidences a strong disinterest, and topics for which URN has no evidence of the user's level of interest.
- *URN must provide an improved representation of Usenet articles.* The effectiveness of current information management systems for Usenet is vastly circumscribed by the poor quality of their representations. This poor quality results from two features. First, representations at the level of newsgroups are too coarse. For example, comp.software-eng is too coarse a representation of a user's interests for effective assessment of individual article relevancy. Second, representations at level of individual articles (i.e. based upon the Subject line or References line) is frequently inadequate or wrong. For example, subject lines frequently do not fully summarize the contents, or even misrepresent them entirely.
- *URN must introduce minimal overhead.* An information management system that introduces substantial additional overhead beyond that which Usenet already incurs will not be successful. Any "investment" the user makes in terms of URN overhead must "pay off" directly in terms of improved Usenet information access.
- *URN must exploit the power of collaboration.* The preceding requirements create a dilemma: URN must provide new, explicit representations for users and information, yet do so without introducing excessive overhead. We believe collaboration is the only effective means to increase the representational quality and expressiveness of a Usenet information management system without introducing excessive overhead on individual users.

Design

To satisfy these requirements, the design of URN incorporates three intertwined design features. First, URN provides a weighting mechanism that explicitly represents both the level and confidence associated with interests. Second, URN provides a simple voting mechanism as part of its navigation facilities that enables users to express their interest in an article without incurring any additional overhead. Third, URN provides a method for users to improve the quality of representation of individual articles by editing a default list of keywords generated automatically for each article by URN. Each of these design features are discussed in more detail below.

Weighting Functions Conceptually, a weighting function takes a feature of an article and a user as input, and returns an integer value representing the level of interest that the user has displayed in that feature. Highly positive values indicate that the user has expresses highly predictable interest in articles with this feature, while highly negative values indicate the reverse. URN might represent our hypothetical Apple lover with one weighting function that assigns a moderately negative weight to articles related to IBM, and another

weighting function that assigns a high positive value to articles related to Apple.

Given a set of features associated with an article, a prediction of the user's interest in an article can be made by simply summing up the values returned by the weighting function for each feature. The user can then order articles by interest level ("weight" is somewhat of a misnomer, considering that articles with high weight rise to the top).

For example, given the weights described above, an article discussing an Apple/IBM partnership would get a low positive rating because the moderate negative weight associated with IBM would be added to the high positive weight associated with Apple. Because conflicting weights indicate representational uncertainty about the relevancy of an article to a user, negative weighting functions make a smaller relative contribution than equivalently positive weighting functions to the total value attributed to an article. This means that URN tends to give higher rankings to articles for which it has been provided with "mixed messages".

Voting Article weighting overcomes one of the problems associated with current Usenet information practice, since it explicitly represents both interest and non-interest at the article level. However, manual generation and maintainance of these values would be error-prone and introduce substantial new overhead. For this reason, URN completely automates the generation and maintenance of weighting functions based upon a simple voting mechanism. To indicate that they are finished with an article, users press one of three keys indicating that the article was: interesting, ambivalent, or uninteresting. The design of explicit voting requires a careful balance because if users are asked to do too much rating, they will become annoyed and stop giving useful ratings. While some interest level information might be inferred indirectly from user actions (such as not completely displaying the entire article before moving on to the next one), the more simple strategy of explicit voting appears more robust. We believe that our three level rating system presents a low enough overheard that that users will make use of it.

The next section on the implementation of URN describes in detail how votes are transformed into weighting functions.

Collaborative Article Representation A second problem introduced by weighting functions is the requirement for each article to be represented in terms of a set of features. As noted previously, a fundamental problem with Usenet is that it encompasses a virtually unbounded set of domains, and the structure imposed upon individual articles is insufficient for both precise and accurate representation of article content. To resolve this, URN generates a default list of keywords to be associated with each article, which can then be edited by any user to better conform to the true content of an article. This keyword-based representation of article content is shared by all users, which means that the overhead of its construction and maintenance is also shared among all users.

Having now overviewed the essential requirements and major design features of URN, the next section presents details of its implementation.

Implementation

URN is implemented using Egret, a Unix/X-window environment for the implementation of exploratory collaborative hypertext applications [6, 7]. Egret consists of a 20 KLOC server process written in C++, which communicates via TCP/IP to Lucid Emacs clients extended with 15 KLOC of Lisp. The URN application specializes Egret to Usenet with approximately four additional KLOC of Lisp.

Inputting Articles URN keeps its own database of Usenet articles. Articles are read into the URN database periodically by an agent process via the NNTP protocol [8]. As it reads in each article, keywords are extracted from the header of the article. The header fields Subject, Summary, Author, and Keywords from the original article are parsed into separate words. Next "noise" words (such as "the", "or", "and") are removed and the words are converted to lower case. These words are then stored with the article, and are called the article's keywords. We keep track of how many unique keywords are in the database, and the frequency of each keyword. Articles in a thread contain hypertext links to their neighbors.

User Interface Once there are articles in the database, users connect to the database through an URN client. After connection, they are shown a list of all the articles in the database, sorted in descending order by weight (see Figure 1). Users click on any article from the list to retrieve and display it. Articles are displayed similarly to other news readers: the header lines are on the top, followed by the list of hyperlinks and keywords, and then the body of the article (see Figure 2).

From any article, the user can click on any of the highlighted links to move to the related article. The user may also select any keyword in the Keywords field and delete it, or select text from the body of the article and add it to the Keywords field (Free-form entry of keywords not found in the article is also supported.) In this way, users collaboratively build a consensual representation of the contents of each article. In fact, one user may vote on an article, and later on another user may change the keywords associated with that article. URN will then recalculate the weighting functions associated with the first user to correspond to the improved representation of the article.

After reading the article, the user votes on the article as either: interesting, ambivalent, or uninteresting. This vote is recorded, and the user goes on to the next article in this thread. When users finish reading articles, they disconnect from the URN database to end their URN session.

Keyword Agent After users vote on articles, a background process called the Keyword Agent takes their votes and turns them into weights. First, the agent reads a list of all keywords that have been manually added by users. The bodies of all articles in the database are then scanned for these words, and if any are found then those keywords are "promoted" to the Keywords field for the article it was found in. User keywords are treated this way because URN assumes that user-added keywords are more relevant than automatically-added keywords, and less likely to be bogus. The net effect is that if a user adds a keyword to an article, then the keyword is added to all other articles in the database containing that keyword

```
sbuff: Unread-Article-Selector
File  Edit  Buffers  URN Commands  Help

                          URN Unread Article Selector
                          ===========================
                           Sun Feb  6 16:30:22 1994

 ID     WT    Author Name             Subject
-------------------------------------------------------------------------------
 1006   840   Christophe Debou        Re: C coding standard checks?
>448    314   Corin Gurr              Re: What ARE Prolog's advantages? (was: +
 1010   314   Corin Gurr              Re: What ARE Prolog's advantages? (was: +
 168    268   Steve Scheuber          Omaha SPIN Searching for Guest Speakers
 1028   263   Tim Dugan               Re: Ada etc.
 414    263   Johan Bengtsson         Re: Ada etc.
 418    263   Johan Bengtsson         Re: ADA etc.
 972    170   Cynthia Mooney          Searching desperately for CASE tools (UN+
 1034   115   Graham Perkins          Re: ON MOVING TO LEVEL 3...
 178    0     Peter Mutsaers          Re: ADA ETC.
 366    0     Erland Sommarskog       Re: Ada etc.
 374    0     Norm Young              Re: Teaching algebraic specification
 386    0     Peter Mutsaers          Re: ADA etc.
 388    0     ryang@ryang2.mese.com   Re: ADA ETC.
 416    0     Frank van Harmelen      CfP: Workshop on Formal Spec Methods for+
 1012   0     Martin Zandbergen       Hatley&Pirbhai -- QA -- MIL
 1024   0     Robert Kitzberger       Re: SWE Nightmare Cartoons
 402    -2    Lars-Henrik Eriksson    Re: What ARE Prolog's advantages? (was: +
 1044   -84   Robert L. Hays          REQ: Mac tools for static analysis?
 188    -84   chia-ming wang          Help on evaluating software development +
 380    -84   Robert Kitzberger       Re: ADA ETC.
 430    -84   Gregory Aharonian       Review of testimony at Patent Office hea+
 980    -105  Bob Gorman              Re: Version Control
 292    -105  Robert Scott            Version Control
--**-rbrewer: Unread-Article-Selector      (sbuff)----Top----------------------
Mark set
```

Figure 1: The URN Unread Article Selector. The second column displays the weights assigned to each article by URN, based upon the votes applied to similar articles by this user in the past. Weights of 0 typically indicate that URN does not have any information about the user's interests relevant to assessing the article.

in their body.

Next, the Keyword Agent uses each user's votes to generate their weighting functions. URN's weighting functions are currently quite simple: a keyword and its associated weight. For each user of the system, URN assembles a list of all the articles that they voted as interesting. URN then builds a list of all of the keywords from those articles, noting the frequency of each keyword. This is called the "goodlist". URN does the same thing for articles that were voted as uninteresting, resulting in the "badlist". URN then compares the two lists and eliminates any keyword that appears in both lists. For each keyword in the goodlist, URN computes a weight using the following formula:

$$W_g = f_l \cdot \frac{1}{f_g}$$

where W_g is the weight computed, f_l is the local frequency of this keywords, and f_g is the global frequency of this keyword. The local frequency of a keyword is simply the number of times it appeared in the goodlist. The global frequency of a keyword is the number of times it appeared in the whole database. By taking the product of the local frequency with the inverse of the global frequency, URN generates higher weights for rare words, and lower weights for more ubiquitous words [1].

For each keyword in the badlist, URN computes a weight using the following formula:

$$W_b = -f_l \cdot \frac{1}{f_g} \cdot \frac{8}{10}$$

where the variables are defined similarly to W_g. Note that bad weights are negative, and that for any given f_l and f_g, W_b will be smaller in magnitude than W_g due to the constant term in the calculation of W_b. As stated previously, URN does this to bias the system toward presenting irrelevant articles with false positive weights rather than missing relevant articles with false negative weights.

Importantly, all the weighting functions are dynamic: they are recomputed each time the Keyword Agent runs. This allows the weights of articles to change over time as user interests change or new information about the user is provided.

Once all individual weighting functions are generated, the Keyword Agent computes the new weights of all articles in the database for each user. For each article, the agent compares the list of keywords for that article to the list of weighting functions for a user. If there is a match, then that weighting function's weight is added to that user's weight for the article.

Other Usenet Readers

Many different programs have been written to read Usenet. What follows is a brief survey of some that are related to URN's goals.

trn, GNUS, etc. These are the standard programs used by most Usenet users. They allow subscription to newsgroups

```
sbuff: <rlk.759983938@bonnie>
File  Edit  Buffers  URN Commands  Help
Message-ID: <rlk.759983938@bonnie>
Newsgroups: comp.software-eng
Author Name: Robert Kitzberger
Author Email: rlk@bonnie.Rational.COM
Organization: Rational
Date: Mon, 31 Jan 1994 02:38:58 GMT
Subject: Re: ADA ETC.
Distribution:

Classification:

Keywords: ("ada" "robert kitzberger" "programmer-as-artist" "software" "reuse")

Followup Links:

Body:
muts@compi.hobby.nl (Peter Mutsaers) writes:

>Yes, I have used ADA, and yes, in my company many have and still do.
>I've seen the results of this mentality: people who made functional
>requirements, spent a lot of time on design, and then thought the
>coding would be a dull and mechanic task. Such projects have utterly
>failed. Maybe this can be done for less complex software, but for
>complex software on unknown territory, a lack of feeling with the real
>code and implementation is deadly.

You've described two extremes of software development: endless design
and the programmer-as-artist hacking approach.   To toss a bit of
obligatory software engineering content into this thread: we stress
an approach that focuses on iterations of analysis/design/develop.
Each iteration happens rather quickly.  Possible risk areas are identified
--%%-URN: <rlk.759983938@bonnie>      (URN)----Top-----------------------------
Mark set
```

Figure 2: An Article Displayed with URN. Articles are displayed along with a field containing a list of keywords representing the collaboratively built, consensual representation of the content of this article.

and threads are explicitly represented. Typically a user will be shown a list of all the threads from a group, and the user can select any number to be read. The only filtering technique provided is kill files. There is primitive support for the automated creation of kill files (i.e. a user may ask to have the Subject line from the article they are reading deposited in their kill file). There is no support for signalling articles as especially interesting, nor is there any kind of collaboration support.

strn. "Scan Threaded Read News" is an enhanced version of trn. It adds the concept of virtual newsgroups which are categorizations of newsgroups by certain search criteria, and assigning "scores" to articles by patterns (similar to URN's weighting functions). Its scores can only be based on the header lines of an article, and they must be created manually. There is some collaborative potential available through sharing of ones' virtual newsgroup index files.

INFOSCOPE. This newsreading tool allows the creation of virtual newsgroups via filtering [3, 11]. The filters are generated by background agents that monitor users' activity and display their findings to the user, who can either accept or reject them. The filtering is performed on individual users and INFOSCOPE does not provide any collaboration support. The filters are a subset of boolean logic, and they only search header lines. It also provides a graphical user interface for browsing through the Usenet hierarchy. Because it derives all its information about articles through the header lines, it cannot determine that a thread's contents have changed from the actual Subject line.

Tapestry. This collaborative document filtering tool [4] contains a complex query language (TQL) which users can use to write their own queries. Filters written in TQL can access "annotations" or "endorsements" created by other users in order to filter a message. These endorsements are similar to votes in URN, and allow for "virtual moderation" of newsgroups. However, filters must all be created manually by the user using TQL. Articles are only prioritized for display in the last step of the Tapestry process using a email client program that does not have access to the full TQL language.

EXPERIENCES WITH URN

We have discussed the techniques we believe are necessary to improve the Usenet reading experience in the previous section. The success of this method depends upon on both URN's ability to create meaningful weighting functions based on users' votes, and users' ability to create a meaningful representation through addition and subtraction of keywords. Our experiment was designed to test the hypothesis that using only these votes and keyword manipulations, URN can generate weighting functions that accurately represent the relevance of an article to a particular user. First we explain the method we used to perform the experiment. Next we present the quantitative results from the experiment. Finally we discuss what conclusions we can draw from these results.

Method

The experimental evaluation of URN consisted of a two week usage of the system with articles input from a single newsgroup. Six members of the CSDL group were asked to use URN on a regular basis to read the newsgroup "comp.software-eng". If they were previously reading this newsgroup with other newsreading software, they were asked

to unsubscribe from the newsgroup for the duration of the usage. Figure 3 shows the experimental statistics.

Number of users	6
Duration of experiment	10 days
Number of articles input	175
Total time spent by all users	13.5 hrs
Size of body text	343 Kbytes

Figure 3: Statistics from URN experiment.

Users were instructed to read articles using URN, and to vote on each article based on how interesting it was to them personally. They were also asked to add or delete keywords from articles when they felt it was appropriate. In order to gain better insight into URN's weighting function generation capabilities, users were asked to read all articles, even if they had very large negative weights. This is quite different from 'normal' usage where users would probably mark all articles below a certain threshold weight as read automatically.

During the experimental period, extensive data was collected on the subjects' interaction with the system. The data was collected using EGRET's customizable metrics-gathering facilities. Whenever a user performed a semantically interesting action, URN recorded what action occurred, who performed it, what entity it was performed on, and the time that it occurred. This data stream is then saved to the database at the end of each session. Our analysis is based primarily on these metrics.

Results

We will discuss two of the different kinds of events we recorded: voting events, and keyword manipulation events.

Votes and Weights Each time a user voted on an article, URN recorded several pieces of information: the user's vote, the article voted on, the type of vote (interesting, ambivalent, uninteresting), the weight of article at that time, and the rank of the article in the Selector at that time.

This information can be used to analyze URN's ability to adaptively generate weighting functions. We calculated the average weight of an article voted as interesting for each two day period, combining the votes from all six users (see Figure 4). Note: articles that were voted on but had a weight of zero have been excluded from this average because they represent articles for which URN had no interest-level prediction.

Due to the relatively small number of data points from this experiment, there is considerable variability in the results. In particular only a handful of "interesting" votes occurred on days 9 and 10, resulting in the downward trend in the graph. Apart from this anomoly, one can clearly see a trend of articles voted as interesting rising in weight, and articles voted as uninteresting falling in weight.

Figure 5 shows a more complete view of one user's votes. The graph shows the weights of articles voted interesting and uninteresting by one user on a per-session basis. Once again, articles voted on which had zero weight are not displayed. We can see the weights diverge from zero over time causing interesting articles to get higher weights and uninteresting articles to get lower weights. Each data point in this scatter plot may represent multiple articles, if they had the same weight and were voted upon in the same day.

Unfortunately, not all of the users graphs looked as convincing. In some cases weights fluctuated over time, and in some sessions there were articles with large positive weights that were voted uninteresting, and vice versa. One reason for these cross-overs is the small number of data points available. In some sessions, only one article was voted as interesting which can skew the results for that session. The other problem, which we discuss later on, is the inability for users to indicate whether they are voting on whether the subject area of an article is interesting to them, or whether the quality of the article is worthy of their interest.

Keyword Manipulation Keyword manipulation metrics were also recorded. Whenever a user chose to add or delete a keyword, the article and keyword in question were recorded. The keyword statistics can be seen in Figure 6. As this table illustrates, about 10% of the keywords in the system were changed by the user. One concern raised about our shared keyword manipulation scheme was whether or not users would accidentally or purposefully delete keywords added by other users, or re-add keywords deleted by other users. During the experiment, only one user deleted a keyword created by another. In this case, the first user had added a keyword containing an extra space, and the second user merely fixed the error by deleting the incorrect keyword and re-adding it without the extra space.

Number of total distinct keywords	782
Number of keywords added by users	33
Number of keywords deleted by users	53

Figure 6: URN Keyword Statistics

Discussion

This experimental evaluation of URN raised several interesting research questions. The most obvious and pressing is the distinction between the category an article belongs to and the overall quality of an article. These are two orthogonal characteristics of an article, but in URN their representation is merged. While URN keywords primarily specify the content of the article, they also contain the author's name by default. Therefore the keywords for an article in URN represent to some degree both the category and the quality of an article.

This merged representation causes problems when users vote on articles. They must condense their interest in both the category and the quality of an article into a single value. For example, during the experiment one user read an article which was an appropriate, high-quality response to a posting that was unrelated to the newsgroup. The user was faced with a conundrum: should the user vote on the article as uninteresting and thereby assign an unfavorable weight to the author of a high-quality article, or should the user vote on the article as interesting and risk seeing more articles on this uninter-

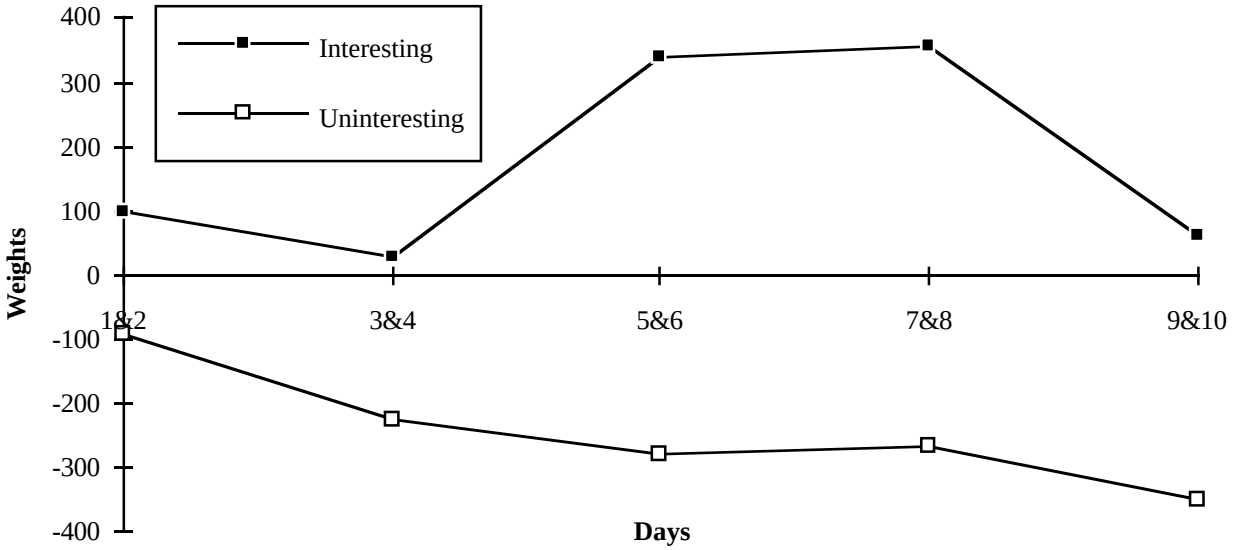

Figure 4: Weight of an article voted as interesting averaged over all users vs. time. This graph combines together approximately 330 votes.

esting topic? The opposite situation also occurs when low-quality posts are made on high-interest subjects. This was seen clearly in the weight/vote plots for users: sometimes articles with large negative weight were voted interesting, and vice versa. In addition, users were not explicitly told whether their votes were supposed to reflect the category or the quality of the article, which led to different users interpreting the intent of voting in different ways!

An important lesson learned from our experiment is that a collaborative system for Usenet must make a distinction between these two characteristics. One possible way is to record two votes for each article: one for the relevance of the category to the user, and one for the user's assessment of the quality of the article.

The current version of URN makes no effort to preserve the thread structure of articles at the selection level; all articles are presented in weight rank order regardless of their thread-level position. During the experiment, one user found that the article with the highest weight was actually a reply to the article with the second-highest weight. This raises interesting issues regarding how the weights of articles in a thread should be combined into a weight for a whole thread. Should the articles with the highest weight in a thread be presented first, or should the thread be presented in chronological order? There is a presentational tension between seeing the highest rated articles and following the logical flow of the conversation.

URN's keyword manipulation system is based on the assumption that that the collaborative group using it have fairly similar interests and views, so that they can come to a consensus as to the set of keywords that describe an article. However, even in small like-minded groups, users may have widely varying representations about the structure of a textual artifact [12]. In this usage we did not encounter any serious problems, but it must be considered in any longer term experiment.

Users expressed a desire to be able to give URN a wider range of feedback on an article. Often when using the system, users wished that they could indicate a stronger preference than the binary interesting or uninteresting provided. Research in psychology suggests that subjects can give meaningful feedback with up to a 7 point rating scale [9], which could easily be implemented in a future version of URN.

CONCLUSIONS AND FUTURE DIRECTIONS

This paper presents URN, a system for collaborative classification and evaluation of Usenet. It combines a collaboratively built representation for the keywords associated with an article with an adaptive interface that prioritizes articles based on votes on previous articles. The paper presents results of a two week trial of our system, providing quantitative evidence to support our claim that URN's weighting functions can represent user's reading interests. Now we discuss our plans for URN in the future.

Our first future direction is to gather more experimental data on URN over a longer time frame. With such data, we can perform more statistically interesting analyses of URN usage, and provide higher quality evidence for the strengths and limitations of this approach. We intend to continue experimenting with URN throughout 1994.

Second, users suggested many improvements to the weighting function mechanism. One suggestion is the ability for URN to provide users with direct access to their weighting functions. On many occasions users wished that they could add a weighting function to their profile directly because they were sure that they were interested or uninterested in a particular keyword. Users also desired better control over keywords, such as the ability to create synonyms or select from a menu of keywords. Finally, users also desired enhanced collaborative capabilities, such as the ability to recommend a specific article to another individual.

Third, URN represents only an initial step in our work toward

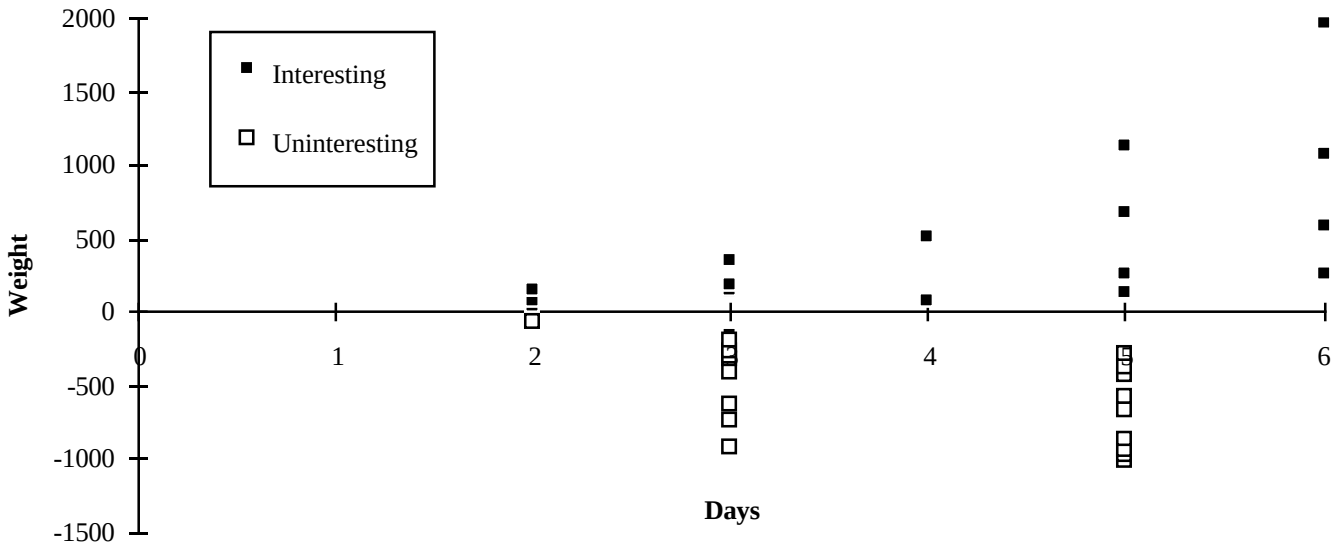

Figure 5: A scatter plot of one user's voting behavior over the experimental period. Over time, interesting articles were more positively weighted, while uninteresting articles were more negatively weighted.

enhanced information access in Usenet. Our medium-term goal is to support a process we call *knowledge condensation*, in which information obtained from Usenet is not simply annotated with keywords in order to support evaluation, but actively restructured with the collaborative addition of hypertext links to other information of interest saved from prior postings. In this way, a group of users can incrementally and collaboratively build and restructure a richly interlinked knowledge base of information about a common topic of interest, and share this knowledge base with others via Usenet.

ACKNOWLEDGMENTS

We would like to thank the other members of Collaborative Software Development Lab: Rosemary Andrada, Carleton Moore, Danu Tjahjono, and Dadong Wan for their assistance in preparing this manuscript as well as in the development of Egret[7]. Robert Brewer would also like to thank Yuka Nagashima for her help in reviewing this document. Support for this research was partially provided by the National Science Foundation Research Initiation Award CCR-9110861.

REFERENCES

1. Nicholas J. Belkin and W. Bruce Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38, dec 1992.

2. UUNET Communications. Total traffic through uunet for the last 2 weeks. periodic postings to the Usenet newsgroup news.lists, January 1994.

3. Gerhard Fischer and Curt Stevens. Information access in complex, poorly structured information spaces. In *Human Factors in Computing Systems CHI'91 Conference Proceedings*, pages 63–70, apr 1991.

4. David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave and information tapestry. *Communications of the ACM*, 35(12):61–70, dec 1992.

5. United States Government. The national information infrastructure: Agenda for action. Available from NTIA NII Office, 15th Street and Constitution Avenue, Washington, D.C. 20230, or electronically via ftp to enh.nist.gov, 1993.

6. Philip M. Johnson. The EGRET project: Exploring open, evolutionary, and emergent collaborative systems. In *Proceedings of the 1991 ECSCW Developer's Workshop*, September 1991.

7. Philip M. Johnson. Supporting exploratory CSCW with the EGRET framework. In *Proceedings of the 1992 Conference on Computer Supported Cooperative Work*, November 1992.

8. Brian Kantor and Phil Lapsley. Network news transfer protocol: A proposed standard for the stream-based transmission of news. Request for Comments 977, Internet Network Information Center, available via ftp to site ftp.nisc.sri.com, February 1986.

9. G. A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, pages 81–97, 1956.

10. Brian Reid. Usenet statisics. period postings to the Usenet newsgroup news.lists, mar 1993.

11. Curt Stevens. Automating the creation of information filters. *Communications of the ACM*, 35(12):48, dec 1992.

12. Dadong Wan. *CLARE: A Computer-Supported Collaborative Learning Environment Based on the Thematic Structure of Scientific Text*. PhD thesis, University of Hawaii, Department of Information and Computer Sciences, 1994.