Department for Computer and Systems Sciences

Stockholm University
Kungliga Tekniska Högskolan

Issues when designing filters in messaging systems

By Jacob Palme, Jussi Karlgren and Daniel Pargman

Abstract: The increasing size of messaging communities increases the risk of information overload, especially when group communication tools like mailing lists or asynchronous conferencing systems (like Usenet News) are used. Future messaging systems will require more capable filters to aid users in the selection of what to read. The increasing use of networks by non-computer professionals requires filters, that are easier to use and manage than most filtering software today. Filters might use evaluations of messages made by certain users as an aid to filtering these messages for other users.

Keywords: Electronic Mail, Message Handling Systems, MHS, Computer conferencing, Bulletin Board systems, Filters, Mail filtering, Netnews, Usenet News, Information Retrieval Systems.

Author's personal address: Skeppargatan 73, S-115 30 Stockholm, Sweden. Phone: +46-8-16 16 67. Internet mail: jpalme@dsv.su.se.

University address: Department of Computer and Systems Sciences, Stockholm University and Kungliga Tekniska Högskolan, Electrum 230, S-164 40 Kista, Sweden. Phone: +46-8-16 20 00.

Gopher server: gopher://dsv.su.se

World Wide Web server: http://www.dsv.su.se

Why filters are needed

Much of the messaging in the Internet today consists of messages sent to many recipients through mailing lists, distribution lists, bulletin boards, asynchronous computer conferences, newsgroups, etc. In the rest of this paper, the word activity will be used as a common term for all these tools for sending messages to groups of recipients.

Common to all these functions is that the originator of a message need only give the name of one recipient, the name of the activity. The messaging network will then distribute the message to each of the members of the activity, with no extra effort for the originator. The average effort of writing a message is about four minutes, and the average effort of reading a message is about half a minute [Palme 1981], so if there are more than about eight recipients to a message, the total reading time is larger than the total writing time, and if there are hundreds or thousands of recipients, the total reading time caused by the originator is many times larger than his effort in writing the message.

Because of this, members of activities will easily become overloaded with messages [Denning 1982, Palme 1984, Hiltz and Turoff 1985, Malone 1987]. Filters are tools to help them reduce and organise the incoming messages. Filters are, for most users, more important for group messages (messages sent to activities) than for individually addressed mail. Future messaging software for the Internet can be expected to employ more advanced and user-friendly filtering functions than today, in order to support larger discussion groups and less computer-specialist users.

Action caused by filters

Filters will recognise certain incoming messages before they are shown to the recipient, and cause different actions to be performed on them. Typical such actions can be:

- To delete the message without reading it.
- To forward the message automatically to some other e-mail address.
- To cause the message to be automatically processed by some particular software.

  This software might for example summarise messages of a certain kind. Instead of getting each message separately, the user will, at a certain time or at regular intervals, get a report with a summary of several messages. One example of such a summary might be a count of vote or a summary of delivery notifications.

  Such special messages are often called *notifications*, and special handling of notifications is often built into e-mail software, one could say that such systems have implicit filters for this particular purpose.
- To save the message (in a mail data base, or in a separate file) for reference, but not show it to the user as a new message.
- To present the messages in a certain order.
- To sort the message into a folder, so that the user can read new messages one folder at a time according to his priorities.

  Note that sorting messages into a separate folder for each activity is a basic function of all *asynchronous conferencing systems* [Palme 1984, Hiltz and Turoff

1985], like Usenet News [Horton-Adams 1987], EIES [Turoff 1991 and Whitescarver 1987] and KOM [Palme and Tholerus 1992]. One could thus say that conference systems and bulletin board systems have implicit filters built into them, sorting messages into folders, one for each activity. Many such systems also filter on *topics* or *conversations* as a substructure of activities.

Most filtering software organises the filter for a user as a sequence of selection conditions, which are applied in a certain order. If a message is selected by one of the conditions, the condition may specify whether the message should be processed by the rest of the conditions or not [Berg 1993], [Taylor 1987], [Pollock 1988] [Wyle 1992]. In otther systems, the order in which the filter rules are specified will not affect the outcome of the filtering [Pargman 1994A]. This is probably easier to understand and manage for users. The effect, which the order of the filter conditions has, on the outcome of the filtering, is often confusing and difficult to grasp. Filtering systems which do not depend on ordering are however more complex to program, since they may have to scan the filter rules more than once. One filter rule might for example set a variable “scientific text” on a message, and another filtering rule might use this setting.

Attributes used in selectors

Filter selectors typically select on attributes like originator, activity or word occurrences in the subject line or in the text of the message. Selecting on the subject line is particularly common, since a series of messages, that are directly and indirectly, replies to each other, often has the same subject line (sometimes with a prefixed "Re: "). Such a series is called a conversation. Thus, selecting on the subject is in reality used to select or deselect entire conversations.

Using word occurrence in the subject and text can for example be used by a user who participates in an activity for Macintosh users, but who is only interested in messages about the Nisus editor or only in messages about computer games.

The time flow when a message was written can be used for filtering in several ways:

- Old messages, which a user has not had time to read for several weeks, may be discarded.
- A user may discard messages, if too much is written during a certain time period in a certain activity.
- The opposite is also possible, the sudden increase of messages in a activity may cause a user to want to see what is happening.

Other attributes that may be useful for selection are textual criteria like the length of a the message, or genre-specific variation of sentence length, word length and other easily calculable text metrics [Karlgren 1994B]. It is in this way possible, to categorise the style in a message as scientific, journalistic or personal, and this may be used for selection.

Semantic analysis of the text of a message might provide better filtering, for example by avoiding a message about “non-Windows user interfaces” for a user who has “Windows” as a search criterion, or by avoiding a message about “computer architecture” for a user who wants to find messages about house-building architecture. However, present knowledge on how to make semantic analysis of text may not be enough to allow acceptable selection quality.

Filters may have problems with conversations

By *conversation* is in this paper meant a set of messages, which are directly or indirectly linked by In-reply-to:, References: and Obsoletes: links, see figure 1.

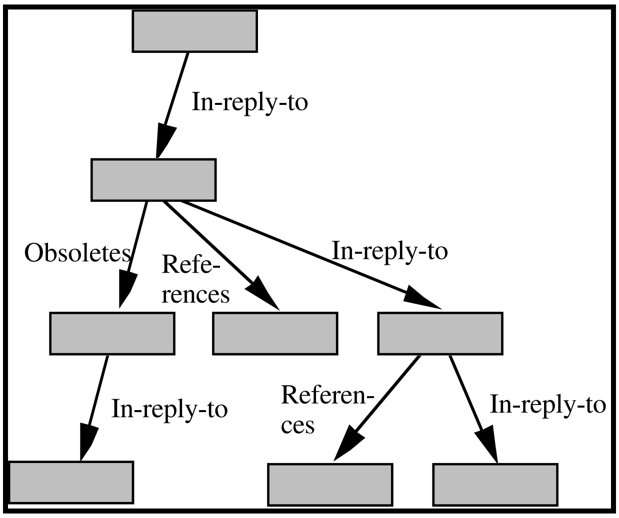

Figure 1: Example of a conversation

A disadvantage with filters is that they may provide a somewhat random selection out of a series of connected messages, where the user may prefer to read all or none of them. More intelligent filters might be able to handle this, by supplying all the messages in such a chain, even if the keyword only matches some of them. When one message in a conversation is found by a filter, the user may wish to see the message which started this conversation in order to decide whether to read the rest of the conversation.

Unfortunately, the *Message-ID, In-Reply-To* and *References* heading fields are not used consistently enough in Internet mail to support good recognition of conversations. Possible, an intelligent filter might use quoted passages (often marked by "> " in the left margin) to recognise conversation links to other messages.

The *Message-ID, In-Reply-To* and *References* heading fields are only useful if the referenced message has a Message-ID, and some mail systems do not produce Message-ID-s on new messages. Thus, filtering on conversation (and on branches within a conversation) would be greatly aided if Internet made it a stronger requirement that all mailers should provide Message-ID-s to messages.

One solution to this problem would be to produce implicit Message-ID-s on messages, which do not have such ID-s. This is done for example by Listserv.

An important selection criterion is also messages related to other messages written by the recipient. If a user starts a new conversation, or actively participates in a conversation by writing their own messages, that user usually wants to read succeeding messages in the same conversation.

Most existing filter systems do not provide conversation support, except insofar as they can sort on the subject field. The reason for this is not that sorting on conversation is not useful, but that it is technically difficult to implement.

Human editor and filters

In so-called moderated activities, all messages are pre-scanned by a moderator before they are sent out to the activity. The moderator weeds out messages obviously sent to the wrong activity, and can sometimes also select on quality and avoid duplication of the same information in more than one message. (I would prefer to use the term pre-moderated, not moderated, since other kinds of moderation than pre-scanning of messages is also possible.)

Sometimes there are special activities, which contain a selection of the most important messages out of one or more other activities. This selection is done by human editors.

The actions of editors and moderators are also a kind of filtering. Seen from the recipient point of view, the act of a recipient of telling his system that he only wants to read messages in pre-moderated activities is the same as telling his system “give me only messages with an approval attribute from the moderator”. This is thus a filtering on an attribute, which was added by the moderator when accepting it in the pre-moderated activity. Filtering or other decision support systems may be specially designed to aid such editors or moderators in their work.

The advantage with pre-moderated activities is that users are aided by the moderator in filtering out the most interesting or relevant messages. The disadvantage is that it slows down the interactive discussion process. The average time to the first reply is often much less than 24 hours in un-moderated activities, while it may be up to a week in pre-moderated activities.

It would thus be an advantage, if filtering systems could be designed which combine the advantages of pre-moderated and un-moderated activities. For example, moderator information might be used to get a user into a conversation, but after having entered that conversation, the user might choose to read all messages within that conversation without moderator pre-processing. Such systems are not common today. Alternately, users would in some way be allowed to choose between reading only messages selected by the moderator, or also other messages. (As an example, a user filter might say “Give me only messages either approved by the moderator or containing the word sequence ‘nuclear energy’.”.)

Another way of helping users to aid each other would be to allow users to input an evaluation of a message they have read, e.g. on a scale from 0 to 9 with 0=“rubbish” and 9=“extremely interesting”. These values could then be used in filters by other users. This requires a delay in the filtering process, so that filtering is not done until enough evaluations have been collected. Either only late users would be aided by the evaluations from other users, or messages earlier rejected might later appear again as selected messages if positive evaluations arrive on them. This requires that the messaging system stores all messages, also those rejected by the filter, for later possible reappraisal. Such storage is common in conferencing systems, less common in ordinary e-mail systems. An example of a system which reappraises old messages when other users liked them is the Tapestry system [Goldberg 1992]. Systems which use evaluations by one user to influence the filtering for another user are sometimes called “social filtering” or “collective filtering”.

Other actions by other users, such as whether they choose to read the whole message or not, whether they replied to a message or not, might be collected by the filter and use as aid in filtering for other users. An example of such a system is the Tapestry system.

A simple scheme might be to filter only on the average of all evaluations. However, different people have different interests and knowledge, which means that a message, which is very suitable to one person, may not be wanted by another person. It might thus be better if filtering was not based on an average of other people's evaluations, but rather only on evaluations done by other people similar to yourself. If this is done, the system must first be able to find people with similar interests. This might be used by looking at which activities they subscribe to.

There may be a computers and privacy issue in handling these evaluations. Best would be if no human were to directly see evaluations made by other people, since this would probably result in more honest evaluations. The evaluations would thus only be used inside the filtering software.

Will people provide evaluations?

Filtering can only be done on information provided by users. This is one reason why filtering on search keywords explicitly added to messages may not work very well, since it may be difficult to get people to provide keywords on messages, especially good keywords useful for filtering by other people. Studies on the use of keywords for information retrieval also shows that they are surprisingly difficult to use, since different people so often use different terms for the same thing [Foltz 1992]. It would be an advantage if other people than the originator of a message were allowed to add keywords to it, as in the SuperKOM system [Palme and Tholerus 1992].

The same problem occurs with filtering on other people's evaluations. The price which the user has to pay for getting aid from other people’s evaluation might be to supply own evaluations of some messages. A way to get people to provide such evaluations might also be to let the evaluations aid them personally and not only aid other people. For example, an intelligent filtering system might use evaluations to improve the filter rules. The evaluations would tell the filter where it went wrong. The filter might then ask the user why, in order to improve the filter rules. An example of such a system is Infoscope [Fischer 1991].

User-friendliness

Most existing filtering software like Elm [Taylor 1987], Procmail [Berg 1993], MailFilter [Wyle 1992] are not easy to set up and require the user to prepare special control files in a language that is not easy to understand for non-computer specialists [McGough 1994]. It is easy to make a mistake in specifying the filtering conditions. And such a mistake can have disastrous effects, e.g. automatically throwing away important messages. The increasing usage of messaging by people whose speciality is not in the computer area, will create a demand for filtering software that is easier to use. This can be achieved by better user interfaces.

The risk of disastrous mistakes in specifying filtering conditions might become even more of a problem if the filtering software aids the user in producing filters, giving the user less direct control of the filtering conditions. It is important, if people are to use filters, that they are able to trust the filters. One way of achieving this, would be that new filtering conditions would in the beginning not automatically trash messages, but rather put the deselected messages into a list which the user can approve manually.

Only when the user after some time of experience is fully satisfied, the filter might trash messages without user control.

Studies on user behaviour when using filters show that filter conditions set up by users are generally very simple [Mackay 1989]. This means that advanced ways of specifying filters using complex logical language may be more of a disadvantage than an advantage for most users, unless the complexity can be hidden from the user by the user interface [Karlgren 1994C]. Viewing filter rules by frames is usually easier for users than seeing them as logical rules in some programming-like language.

Another way to make it easier for a user to specify filtering conditions, would be to ask the user to input an evaluation on those messages which the user believed were filtered wrong (e.g. a low evaluation on messages accepted by the filter but not wanted by the user). To make this easy, the user interface could be designed so that a single digit, input after reading a message, would be interpreted as an evaluation on a scale from 0 to 9. The filtering program could use this digit plus the message it applies to, to deduct a filtering rule, possibly in co-operation with the user. This could be seen as an expert system, where the filtering program in co-operation with the user builds an expert system data base with filtering conditions for that particular user.

An important issue is whether the filter should act automatically, deriving filtering rules and improving its performance, or whether it should regularly communicate with the user, asking for advice, showing prospective new filtering rules etc. The reason users want filter is to make their life easier, but regular contact between filter and user may increase user control of and user confidence in the filter.

Filters as a replacement for addressing

One interesting idea on filtering was put forward in [Malone 1986, Malone 1987, Terry 1990 and Robinson 1991]. The idea was that the originators of messages need not input any explicit recipient at all of their messages. They could simply send their messages to “anyone”. Each reader would have a filter, which scans all messages sent to “anyone” and selects those of interest to that user. [Terry 1990] says that this would aid both writers, which do not have to specify recipient, and recipients with better selection. However, most experience with this method is from the Information Lens system, which has semistructured messages and thus provides many attributes for selection. The method may not work so well in other message systems which provide less structure on their messages.

Filtering can thus either be applied on the incoming message stream for one particular user (e.g. mail messages and newsgroup items in newsgroups where this user is a member) or to a general data base of available information (e.g. all messages in the Usenet News data base).

Filter in the client or in the server

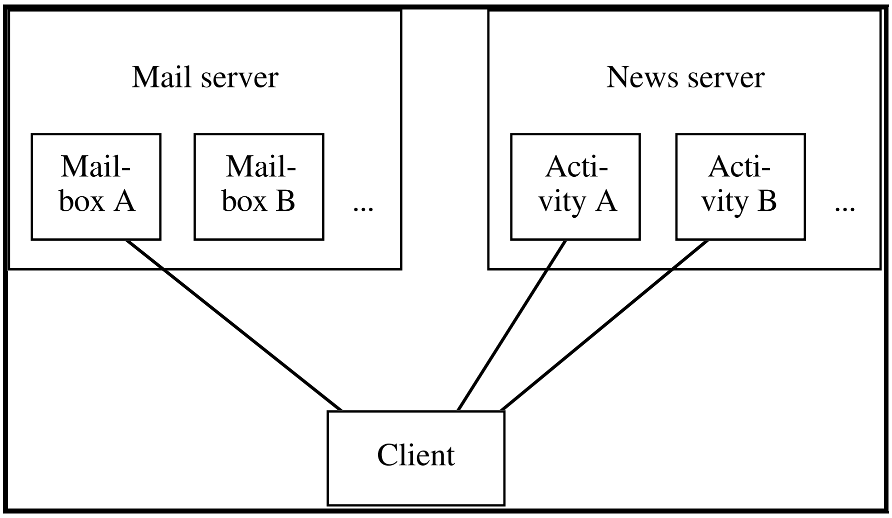

Figure 2: Typical client-server architecture of messaging software

Modern mail and computer conferencing (Netnews) software is more and more often based on a client-server architecture. Each user runs a client program, which accesses a messaging server. For mail, this server usually already has a queue of messages to be sent to this user. For computer conferencing, the server may have such a queue, or it may find the new messages for a user when the user asks for them.

A filter could then either be placed in the client or in the server. The advantage with a filter in the server is that messages, which are to be filtered out, need not be downloaded to the client. This is especially important if the user is waiting at his terminal during the downloading, and if the user is connected to the server via a slow phone line. A server-located filter could filter off-line, so that the filtering result is already available when the client connects. Another advantage with filtering in the server, is that if evaluation by other users are employed in filtering, filtering in the server makes it easier to ensure that the evaluations made by one person is not divulged to another person. With filtering in the client, other users' evaluations would then have to be downloaded to the client software for those using the filter.

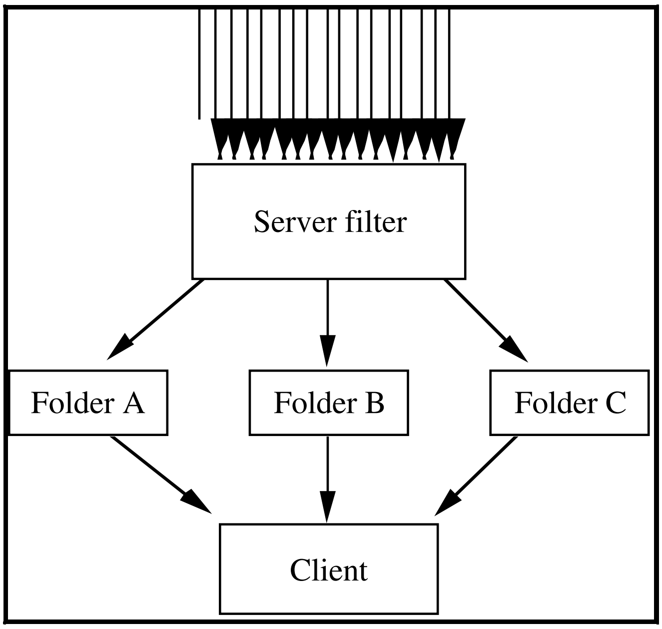

Figure 3: Filter in the server pre-sorting messages for the client

With filtering on all messages in a data base, then filtering in the server becomes even more important. For an implementor, it is however sometimes easier to develop a system with filter in the client.

Filter – Message program interfaces

This section discusses how to interface filters to already existing messaging software.

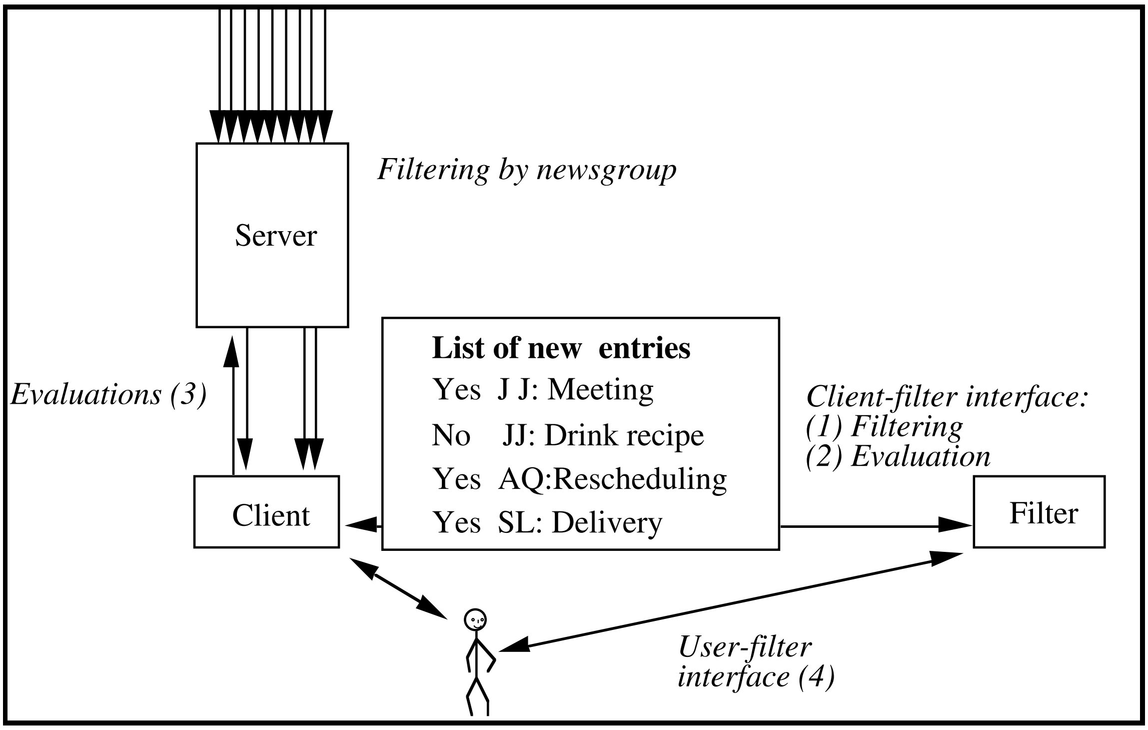

Figure 4: Filtering in the client

Figure 4 shows possible interfaces between a filtering agent and a messaging program with filters in the client. The interfaces (numbers as in the figure) are:

(1) The client sends new messages to the filter agent before showing them to the user, and gets filtering result back.

(2) Evaluations by the user on messages are sent from the client to the filter agent, to teach it to improve its filtering capabilities.

(3) Evaluations made by one user are sent by the client to the server, and back from the server to other user's clients, in order to use the evaluations as guidance when filtering the same message for other recipients.

(4) A direct user interface to the filtering agent is necessary so that the user can review and revise the filtering conditions.

Note that with filter in the client, there might be an advantage, to reduce download time, to download only summary information on the messages first (e.g. sender, date, length, subject) and then filter on this information before deciding to download the whole message. This is of most importance to those users who connect via slow phone lines than for those with high bandwidth local area network connections.

List of new messages in newsgroup A			
Remove the X in front of messages you do not want to read			
X	Elizabeth Wunderba	18 Nov	Prognosis for Hepatitis B
X	John Pearson	18 Nov	Re: Prognosis for Hepatitis B
X	Fred J. Smith	19 Nov	Re: Prognosis for Hepatitis B
X	Mary von Flüger	08:32	We have a shortage of high den
X	Fred B. Nelson	09:07	Problem with the X-ray in room

Figure 5: List of messages for manual filtering by the user

Many existing messaging software products employs a kind of manual filtering of a similar kind. The user is first shown a list of new messages with only one line per message, and then marks which of these messages he wants to read. Only the marked messages need then be downloaded to the client. A filter could in such a system provide suggestions for which messages to be selected by the user, but giving the user the final say on confirming or changing these suggestions.

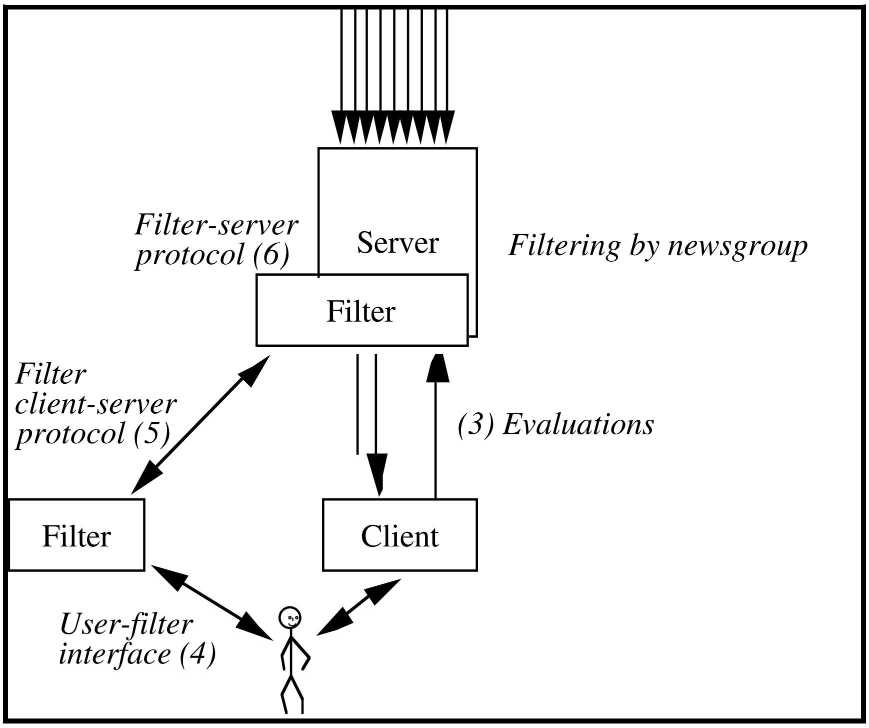

Figure 6: Filtering in the server

Figure 6 shows possible interfaces between a filtering agent and a messaging program with filters in the client. The interfaces (numbers as in the figure) are:

(3) Evaluations by the user are sent to the server to be used both to improve the filter for this user, and as guidance when filtering the same message for other recipients.

(4) A direct user interface to the filtering agent is necessary so that the user can review and revise the filtering conditions.

(5) A filtering client for communication with the user about filtering conditions needs communication with the filtering agent in the server, to download and upload filtering rules and other information.

(6) The filtering agent in the server is interfaced to the server so as to filter messages before they are sent to the user.

Conclusions

The increasing volume of information in computerized discussion groups makes filters more and more necessary. But many users find existing filtering tools too difficult to install and use, and the task of designing well-working filters is complex, involving many important issues such as:

- Should the filter be silent in the background, or in common active communication with its user?
- Should the filter utilize other user's evaluations in filtering?
- Should the filter apply artificial intelligence methods for automatically deriving filtering rules or should it rely on user-specified rules?

There are no simple answers to these questions, and more research in this area is needed.

Current research and acknowledgements

At our university department, we are presently researching the structure of a filtering agent built on the ideas presented here.

The ideas in this paper grew out of the co-operation within this project, in which participated Ann Lantz, Carl Gustav Jansson, Jacob Palme, Daniel Pargman, Fredrik Kilander, Jussi Karlgren, Olle Palmgren, Torgny Tholerus and Yvonne Waern. Other reports from our project are [Lantz 1994], [Karlgren 1994A] and [Pargman 1994A and Pargman 1994B].

References

Note: Some of the referenced papers are available by Gopher in the Internet from URL gopher://dsv.su.se/11/dsv-reports/research-reports.

Berg 1993: Procmail - autonomous mail processor, by Stephen R. van den Berg, WTH-Aachen, Germany. Available from URL ftp://ftp.informatik.rwth-aachen.de/pub/packages/procmail

Denning 1982: Electronic Junk, Communications of the ACM no. 23 vol. 3, March 1982, pp 163-165.

Fischer 1991 et al: Information access in complex, poorly stuctured information spaces, by Fischer, G. & Stevens, C. CHI'91 proceedings, pp 63-70.

Foltz 1992 et al: Personalized information delivery: An analyusis of information filtering methods, by Peter W Foltz and sustan T. Dumais, CACM, DEC 92, Vol. 35, No. 12, pp. 51-60.

Goldberg 1992: Using collaborative filtering to weave an information tapestry, by David Goldberg, David Nichols, Brian M. Oki and Douglas Terry, CACM, December 1992, Vol.35, No.12, pp.61-70.

Hiltz and Turoff 1985: Structuring Computer-mediated Communication Systems to avoid Information Overload, by S.R. Hiltz and M. Turoff. Communications of the ACM, Vol. 28 No 7 July 1985, pp 680-689.

Horton-Adams 1987: Standard for Interchange of Usenet Messages, by M. Horton and R. Adams. Network Working Group RFC1036, December 1987.

Karlgren 1994A et al: An overview of the Intfilter project, by Jussi Karlgren et al. Paper submitted to the CHI'94 conference.

Karlgren 1994B et al: Recognizing Text Genres with Simple Metrics Using Discriminant Analysis, by Jussi Karlgren and Douglass Cutting, in Proceedings of COLING 94, Kyoto, 1994.

Karlgren 1994C et al: The Glass Box User Model for Filtering', by Jussi Karlgren, Kristina Höök, Ann Lantz, Jacob Palme and Daniel Pargman, in Proceedings of the 4th International Conference on User Modeling, Cape Cod, 1994 (Longer version available as SICS Technical Report T94:09).

Lantz 1994: How do experienced users of the system Usenet News select their information. Not yet published working paper.

Mackay 1989 et al:	How do experienced Information Lens users use rules? by Mackay W.E., Malone T.W., Crowston K., Rosenblitt D., Card S.K. and Rao R. Proceedings of the ACM Conference on Human Factors in Computing Systems, Austin, Texas, April 30-May 4 1989.
Malone 1986 et al:	The Information Lens: An Intelligent System for Information Sharing in Organizations, by T.W. Malone, K.R. Grant and F.A. Turbak, CHI'86 proceedings, 1986.
Malone 1987 et al:	Intelligent Information-sharing systems, by Malone, Grant, Turbak, Brobst and Cohen. Communications of the ACM, May 1987, Vol. 30, No. 5, pp 390-402.
McGough 1994:	Filtering Mail FAQ, available from Internet FAQ servers such as http://www.cis.ohio-state.edu/hypertext/faq/usenet/mail/filtering-faq.
Palme 1981:	Experience with the use of the COM computerized conferencing system, DSV, Stockholm University, 1981, re-published 1993, Available via gopher from dsv.su.se.
Palme 1984:	You have 134 Unread Mail! Do You Want To Read Them Now? by Jacob Palme. In Proceedings of the IFIP Wg 6.5 Working Conference on Computer-Based Message Services, 1984.
Palme and Tholerus 1992:	SuperKOM - design considerations for a distributed, highly structured computer conferencing system, by J. Palme and T. Tholerus, in Computer Communications, vol. 15, no. 8, october 1992 pp 509-518. Available via gopher from dsv.su.se.
Palmgren 1994 et al:	GHOSTS - a filter for information streams, by Olle Palmgren, Jussi Karlgren and Daniel Pargman. Submitted as a poster for the CHI'94 conference.
Pargman 1994A:	Creating a Humane Information Flow, by Daniel Pargman. Master’s thesis at Uppsala University, Uppsala, 1994.
Pargman 1994B:	How to create a Humane Information Flow, by Daniel Pargman, Jussi Karlgren, Ann Lantz, Olle Palmgren and Kristina Höök. SICS report.
Pollock 1988:	A rule-based message filtering system, by S. Pollock. ACM transactions on office information systems, Vol 6, No3, July 1988, p.232-254.
Robinson 1991:	Through a lens smartly. By M. Robinson. Byte, May 1991, p177-187.
Taylor 1987:	The Elm Filter System Guide, by Dave Taylor and Syd Weinstein. Available by Gopher from gopher.emr.ca as /public/doc/elm-docs/filter-elm.
Terry 1990:	7 steps to a better mail system, by Douglas B. Terry. Xerox PARC, CSL-90-12, September 1990 and in "Message handling systems and application layer communicaton protocols", P. Schicker and E. Stefferud, Eds., North Holland, 1991 pp.23-33.
Turoff 1991:	Computer-mediated Communication requirements for group support, by Murray Turoff. Journal of organizational computing, volume 1, number 1, 85-113, 1991.

Whitescarver 1987: A Network Environment for Computer Supported Cooperative Work by J. Whitescarver et al, Proceedings of the ACM SIGCOMM '87 Workshop: Frontiers in Computer Communications Technology, ACM Press, 1988, 230-244.

Wyle 1992: A Rule-based Electronic Mail Filter, by M F Wyle, Institute of Technology, Zurich, Switzerland. Available by Gopher from rohan.sdsu.edu.