GHOSTS - A FILTER FOR INFORMATION STREAMS

Olle Palmgren, Jussi Karlgren, Daniel Pargman

Swedish Institute of Computer Science, Box 1263, S-164 28 Kista, Sweden

intfilter@sics.se

ABSTRACT

The GHOSTS system is intended to cut down on information overload when reading usenet news or e-mail. This is done by applying a filter composed of user defined rules to a stream of messages. The aim is to give recipients more control than they typically have today, by giving them better and more varied information on the material they receive, and tools to utilize it.

KEYWORDS

Information filtering, message understanding, conference systems, USENET news, electronic mail

INTRODUCTION - NEW MEDIA

New electronic media such as electronic mail and conference systems are now spreading increasingly fast as they are adopted by corporations and government agencies. They are effective means for fast coordination, dissemination and retrieval of information. As with all new media that replace or supersede older media they have specific advantages, but with their increased usage also some unforseen and unwanted *dis*advantages [1]. The most obvious of the disadvantages is that of intormation overload.

The GHOSTS system is intended to reduce information overload when reading usenet news or e-mail. This is done by applying a *filter* to a stream of messages [2]. GHOSTS is a program package consisting of three programs. Two filters (one for electronic mail and one for Usenet News) and a rule editor. The rule editor is used for managing the rules that control the filter, but it also contains a browser for Usenet News newsgroups.

The users interact only with the rule editor in order to define the behavior of the filter. The filter components are completely transparent to users once they are set up, and the only thing noticeable about their presence is their effect. An important design decision with GHOSTS is that it is system independent and easily portable: users can use the mail and news system they have used previously.

INFORMATION OVERLOAD

Information overload in new electronic media stems from the skewed relationship between the costs of the sender of information and those of the recipient in relation to other media. Once the sender has created information, the cost (in both money and effort) of sending it to many rather than to few recipients is very low. On average, the time to write an electronic message is four minutes, and the time to read it half a minute [3]. If this information is sent to people who have not requested it, or have no need for it, it is they who have to bear the costs of disposing of it. The problem from the recipients' point of view is that electronic media shift virtually all power and initiative from the recipient to the sender. The only action left to the recipient is to refrain from using the information channel at all.

By filter systems we mean software tools that help the user to structure and control an information flow in a manageable way by providing help in the entire spectrum from decision support to autonomous decision making. The aim is to redress the imbalance between sender and recipient caused by new characteristics of electronic media.

GHOSTS has an explicit and editable knowledge representation on several levels. Firstly, as regards the domain knowledge, in the form of a newsgroup browser. The browser is a specialized tool for Usenet News which displays the conference hierarchy and allows for certain actions on the conferences. The simple but surprisingly useful graphical interface makes the several thousand meetings of the Usenet News hierarchy manageable and encourages exploration. Secondly, as regards the system knowledge in the form of simple rule editor and expert system engine for if-then rules that are used to sort and process messages and news texts as the Information Lens and related systems [4]. Thirdly, as regards the domain objects, in the form of support for text analysis tools for efficient data manipulation.

DESIGN PRINCIPLES

Some of the main constraints of this project was the need for ease of extension, portability and reuse of code. The filter system should not be mail or newsreader system specific, but a general agent type assistant to existing systems. This led to the choice of the object-oriented paradigm for the development of the programs. The system has a clean design which is easy to extend. All system specific code – the interface is written using the Motif widget set, in the current implementation – is isolated in a few modules for ease of porting. When the filter was designed some basic building blocks – C++ virtual base classes – were

designed: an expert system, rules and messages. Actual rules and message objects were then derived from the general classes.

Multiple knowledge sources from user studies

The main design principle is to allow users to access multiple knowledge sources that would be too confusing and too many for the individual to cope with. With the aid of an expert-system like knowledge system, much more information can be processed, and messages can be categorized not only by probable content but also by genre, and, tentatively, by quality [5]. Information used by the system is taken from the header fields and the text itself, and includes simple textual data: word occurrence in text or header fields, average word length, average sentence length, part-of-speech information, and other easily obtainable data, as well as data typical of newsgroup messages rather than texts in general: dialog or comment structure that the message belongs in, length of the message, proportion quoted material, length of signature field.

What the genres are in the medium of USENET News conferences is a research question, and we have initiated a study in this area. We are also conducting group discussion studies of experienced users to investigate what other classification parameters and what text typologies user currently are aware of.

THE FILTERS

The filter interacts with a sequential stream of messages. It reads one message at a time from the stream, tries to classify the message and if allowed, perform some action on the message. Example actions are saving the message in a folder, discarding the message, forwarding the message... A research question is what actions are reasonable to allow as a result of rule application. Should a rule be allowed to perform non-undoable actions? When several rules are applicable, what is to be done: should rules be weighted or prioritized?

If the filter fails to classify the message nothing is done with it. The message is passed to its destination undisturbed.

In order to be able to make any judgment at all about the messages the filter has a small expert system. The filter passes the message it reads from the input stream to the expert system and asks it to evaluate the message. The expert system evaluates the messages according to a set of user defined rules. The expert system is also responsible for asking the right actions to apply themselves upon the message.

The filter engine (the expert system at the moment) is constructed to work on a "generalized" model of rules and messages. This makes it possible to construct new rules, and new messages without changing the filter engine. This is implemented using "virtual base classes" and inheritance. The actual rule-classes and message classes inherit an interface from their virtual base classes but provide their own specific behavior. The engine sees the rules and messages as objects of the "virtual base class" type. It does not care about which actual type of rule or message it is working with.

The expert system only knows about the general rules and messages. It knows what they are able to do, but leaves the details of how they do it to themself. This makes it possible to use the same expert system in different filters.

THE RULE EDITOR

The rule editor reads user rule files. It can handle rules for both the news filter and the e-mail filter. The set of rules is displayed in a work area where the user can switch between the two filters at will. The user selects rules for editing by clicking on rule icons, and a rule template edit window is shown.

The rule editor also includes a group browser. This is a tool which displays a graphical representation of the Usenet News group structure where the user can subscribe to groups or enter rules for new group filtering.

INTELLIGENCE - LEARNING NEW RULES

The first step of the IntFilter project is simply empowering users by providing them with more and better structured information. The project will extend to investigating the utility of using learning mechanisms to learn to weight the multiple knowledge sources available. Any “intelligent” learning mechanisms must result in user editable and verifiable knowledge of the same sort that the users themselves have placed in the model, and any intelligent behavior must be phased in incrementally, subject to verification and later editing by the user.

REFERENCES

1 Lee Sproull and Sara Kiesler. 1991. *Connections - new ways to work in the networked organization*, MIT press.

2 Jacob Palme, Jussi Karlgren, and Daniel Pargman. 1993. “Issues when designing filters in messaging systems”, Technical report available via gopher from dsv.su.se at Dept. Comp & Sys. Sci, Stockholm University, Stockholm, submitted to International Networking Conference, Prague.

3 Jacob Palme. 1981. “Experiences with the use of the COM Computerized Conferencing System”, Technical report available via gopher from dsv.su.se at Dept. Comp & Sys. Sci, Stockholm University, Stockholm.

4 Thomas W. Malone, Kenneth R. Grant, Franklin A. Turbak, Stephen A. Brobst and Michael D. Cohen. 1987. ”Intelligent information-sharing systems”, *CACM* 30:5, pp. 390-402.

5 Jussi Karlgren and Douglass Cutting. 1994. “Identifying Text Types Through Simple Metrics”, submission to Coling-94, Kyoto.