A MULTI-AGENT MODEL FOR CONTENT-BASED ELECTRONIC DOCUMENT FILTERING

N. S. PAPASPYROU, C. E. SGOUROPOULOU AND E. S. SKORDALAKIS
National Technical University of Athens
Department of Electrical and Computer Engineering
Division of Computer Science, Software Engineering Laboratory
Polytechnioupoli, 15780 Zografou, Athens, Greece.
Tel. +30-1-7722486, Fax. +30-1-7722519.

A. V. GERBESSIOTIS
Oxford University Computing Laboratory
Parks Rd., Oxford OX1 3QD, UK.

P. LIVADAS
University of Florida, Computer and Information Sciences Department
Gainesville, FL 32611, USA.

1. Introduction

One of the most important tasks in every office is *document management* which includes the subtasks of creating, archiving, retrieving, dispatching, updating and processing documents. In the last decades and in many different ways, there have been attempts to automate these tasks with positive results. Technological advances allow these tasks to be fully automated through the use of computers and specialized software.

Document management is considered as one of the principal applications of computers. Documents comprise 80% of information that is today available in electronic form, in contrast to classic structured databases [1]. Electronic documents are no more a simple analogue of paper documents; they are more dynamic entities in multiple forms and media. They can include information related to their origin and executable code that undertakes their management, something which is not possible in any piece of paper. Documents become the focus of attention when it comes to software development; their easy and user-friendly management is extremely important and new tools are required.

Automation of tasks related to document management has never ceased to be the primary aim of computer science in office environments. The long-term aim is the *paperless office*, whose implementation in the near future still seems unrealizable. In the paperless office, documents will exist in electronic form and be accessed in a way that is easy and “personal” to each user. Many pilot systems have been implemented, studied and evaluated by researchers without significant positive results so far. The most difficult problem today seems to be that of the integration and collaboration of different applications that manage heterogeneous documents.

One solution to the problem of document management that has been proposed recently is the use of *intelligent agents*. An intelligent agent in this context can be perceived as a software entity that mediates between a user and a software system and undertakes tasks that the software system cannot fulfill on its own. The use of an agent as a mediator facilitates and simplifies a user’s job and therefore increases his productivity. Agents provide an elegant solution to the problem of integrating heterogeneous forms of information and incompatible applications for document management and dispatching [2, 3]. Related research has proved that they can be used as a means towards the automation of various tasks that are performed in an office (e.g. management of electronic mail [4] and electronic news [5], scheduling of meetings [6], database and library management, etc.) and in the field of education [7]. Furthermore, a significant part of related literature is concerned with the specification of protocols for communication between intelligent agents, as well as their implementation [8].

One particular subproblem of electronic document management is that of *content-based document filtering*. A category of intelligent agents that is strongly related to this subproblem is that of *filtering agents*. Such agents filter information arriving in the form of electronic documents and present to the user only those that the user considers as interesting, while rejecting useless to the user information. Filtering agents usually co-operate with knowledge bases that contain the user’s filtering criteria. The intelligence of filtering agents is characterized by the fact that users do not need to explicitly specify the filtering criteria. During a training period, a filtering agent automatically learns the filtering criteria of a particular user under his indirect guidance. Given the rapidly increasing quantity of information that is circulated in electronic form, as well as the increasingly specialized content, filtering agents are becoming an extremely useful tool for efficient access to sources of information. In this chapter we suggest a model for content-based electronic document filtering that is based on a multi-agent system.

2. Definition of the problem

The filtering of conventional printed documents is based on their *contents* and not on external characteristics, such as color, texture or number of pages. The problem of *content-based electronic document filtering* is interesting from a researcher’s point of view. It is desirable to develop a system that will automate this process and act as a mediator between the source of information and its human targets. The system should possess adequate intelligence and allow the coding, maintenance and update of the filtering criteria of each of its users. The model that we suggest in the following section attempts to solve this problem. At the same time, we see fit to deal with other related problems, such as *locating* and *collecting* information, its *archiving* and the *exploitation of existing archives*.

One of the basic requirements that a system for automatic filtering of electronic documents is document form *independence*. Total independence may not be feasible; it is at least desirable, however, that users perceive the filtering process in a unified way.

The need for *personalizing* the system for each user results in a model that is very different from those of typical information retrieval systems. The difference lies mainly in the fact that filtering criteria must not stem from an isolated query event, but must result from a period of interaction with the system. The filtering system must be able to cope with long-term changes, based on chains of such events. The issues of *training* and *adaptability* are therefore of great importance also. Besides, the system must be able to selectively forget acquired knowledge, in case the user's interests change. For all these reasons, intelligent agents are an appropriate solution for dealing with all these peculiarities.

In the model's definition, emphasis must be given on the issues of agent *competence* to correctly automate the required tasks and on their *believability*, taking into consideration user psychology and giving the users a feeling of control over the filtering process. The issue of *communication* between agents and users is also very important. The existence of a pleasant and flexible working environment with a user-friendly interface, capable of dynamically adapting to the personal requirements of a user, is an important requirement. So is an agent requirement for *automatic training*, which can be realized in various ways, such as observations of the user's actions, use of examples and counterexamples, evaluation of user feedback, or through direct user guidance.

3. Description of the model

As a solution to the problem of content-based electronic document filtering, we suggest a multi-agent model that mainly consists of four intelligent agents and a knowledge base. A schematic overview of the model is shown in Fig. 1. The part of the figure that is enclosed in the dotted line deals exclusively with the filtering of documents. The rest of the figure deals with auxiliary functions that implement necessary parts of the desired automation.

At the two ends of the suggested model, in the upper-left and lower-right corners, are located respectively a source of information and the user. The model, as has already been mentioned, specifies a system that mediates between these two ends. Key positions in the suggested model are occupied by the four intelligent agents, which are shown in the figure as light bulbs. The agents are collaborating software entities that implement the system's functionalities. They are described in detail in the following paragraphs.

In the description of the suggested model, emphasis has been given on its simplicity, so that the model is well defined and can be checked against its specifications. For this reason, the following two simplifications have been made:

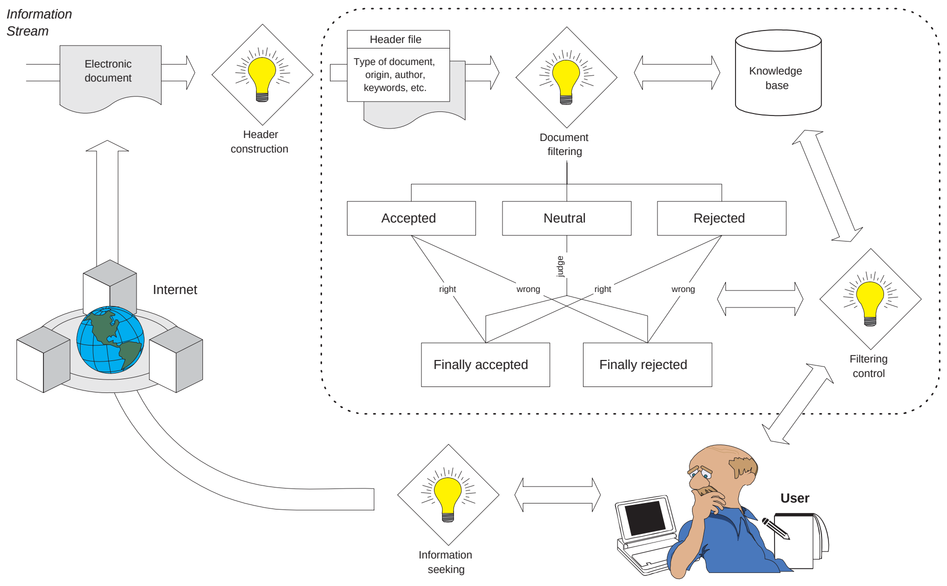
(A1) The model supports a single user and a single category of interests for this user.
(A2) The process of header construction, that will be described in the next section, is not a user-specific process and does not depend on the contents of the knowledge base.

Directions on how to remove these simplifications are discussed in section 5.

3.1. HEADER CONSTRUCTION

In order to deal with the heterogeneity of electronic documents and to simplify the filtering process, it is necessary to introduce an intermediate stage of processing, during

Figure 1: Overview of the model.



which a *header file* is constructed for every electronic document (header files will be also called *headers* for brevity). The format of headers is common for all forms of electronic documents. The task of header construction is similar to the extraction of appropriate *metadata* from electronic documents [9]. In the proposed model, it is performed by the *header construction agent*, which needs not interact with the user. Following this stage, the filtering process is based solely on headers.

The format of the headers must be general enough to describe accurately the key elements of the documents which are useful for filtering. A header contains a set of entries, each one of which consists of a field and its associated value. A *field* can be an arbitrary sequence of letters and numbers, whereas a *value* can be an arbitrary sequence of characters, surrounded by double quotes. An example of a header describing an electronic announcement is given below:

origin	=	“Reuter News Agency”
author	=	“Alan Smith”
title	=	“New explosion in London underground”
date	=	“December 12, 1996, 17:55 GMT”
category	=	“Foreign current events”
keywords	=	“IRA, explosion, underground, terrorism, casualties, statements”
language	=	“English”
wordcount	=	“1340”

The way in which headers are constructed depends a lot on the application area. It is not possible to predefine the names of fields or their values in a unified way that is appropriate for all application areas and document forms.

The automatic construction of headers from heterogeneous electronic documents is a very hard problem and is not expected to be solved in its general form in the near future. Before this happens, computers must be made capable of understanding all forms and media of electronic documents (text, images, sound, video), which is not feasible with the current state of technology. However, satisfactory approximations have been developed for the case of text documents, which comprise a significant percentage of information available in electronic form. Furthermore, some electronic documents are accompanied by descriptive information in textual form. With all this in mind, automatic construction of headers should not be seen as a utopian aim. If the header construction agent is not able to perform its duty, for one reason or another, human intervention is necessary.

Currently, significant research is conducted in the field of *semi-structured data*, aiming at the extraction of information and the automatic classification of electronic documents. So far, promising results have been achieved and the reader is referred to [10, 11, 12, 13] for a summary. The present work differs mainly in two respects: it allows more freedom of form and content in the extraction of information from electronic documents and it uses an intelligent agent with automatic learning capabilities in the extraction process.

3.2. DOCUMENT FILTERING

The filtering of documents in our model is based on a set of subjective criteria that are specified (mostly indirectly) by the user. We assume that the already constructed headers for the new documents that are about to be filtered have been placed in an *arrival area*. Each header is examined by the *filtering agent*, which finally assigns to it a grade of interest, which will subsequently be called *score*.

The score is a real value in the interval $[-1, +1]$. The value $+1$ corresponds to documents that are considered interesting without any doubt and are therefore accepted for presentation to the user, whereas the value -1 corresponds to documents that are rejected with no doubt. Intermediate values represent the agent's degree of confidence, as far as the filtering process is concerned. The value zero corresponds to documents for which the agent cannot make a confident decision.

After the filtering agent comes up with a score for a particular document, based on the document's header and following an algorithm that will be described in detail in §3.4, the agent classifies the document in one of three possible ways, that determine the destination area of each document. *Accepted documents* are those with score near $+1$, *neutral documents* are those with score near 0 and *rejected documents* are those with score near -1. The values for the two thresholds that separate the three areas must be specified directly by the users, according to their needs and to the degree of fidelity that is sought for the classification process.

A necessary property of the filtering agent is the knowledge of its competence. The agent must be able to estimate its critical capabilities, based on the degree of its training and, mostly, on the results of its previous judgements, as is described in detail in §3.5. This property can be reflected in the model by a real value in the interval $[0, 1]$, representing the degree of the agent's *confidence* in itself. A value of 0 means that the agent does not trust itself at all and should therefore not attempt any classification, whereas a value of 1 means that the agent has total confidence in its decisions. In order to take into account this “hesitation” that is displayed by the agent, our model multiplies the score of each document by the degree of confidence. Following that, the filtering agent classifies the documents based on the product of these two values.

3.3. KNOWLEDGE BASE

The knowledge base contains the criteria for the filtering of documents, coded in an appropriate form. These criteria are specified by the user gradually and mostly in an indirect way. The coding of the criteria must be invisible to the user, if we expect the system to be user-friendly. Furthermore, the form of the criteria must be at least as general as the form of the headers, since they will be applied to them.

In the suggested model, filtering criteria are stored in the knowledge base in the form of *rules*. Each rule consists of a condition and a value of interest. Before a rule is applied to a document’s header, it is first examined whether its condition is true. If it is, the score of the document is updated by including the rule’s value of interest, otherwise nothing is done. This process is described in detail in §3.4.

The *condition* in a rule is a logical expression that contains field names, which may appear in the document’s header, as well as constant values. Useful cases of simple conditions aim at checking whther the value of a specific field in a document’s header is equal to (or contains) a specified value. It is also useful to be able to combine simple conditions in more complex ones. The result of applying a rule to a document’s header (if there is one) is to attribute a *value of interest* to the document. Such values are elements of a set which will be called $\mathcal{E}$ in the rest of the chapter. The form and properties of this set’s elements will be described in detail in §3.4.

The description of the representation of the rules in the knowledge base is outside the scope of this model discussion. It is worth mentioning, however, that realistic implementations of the model would give special emphasis on efficiency issues related to the access of the knowledge base. This becomes critical as the volume of information that is stored in the knowledge base increases.

3.4. SCORE CALCULATION

The calculation of the score that corresponds to a document uses as input the document’s header and the knowledge base that contains the required filtering criteria. The score comes up from the application of all rules that are contained in the knowledge base. Each rule whose condition is true for the given document contributes a corresponding value of interest to the total score. The application of all rules may result in a sequence of different interest values, that must be appropriately combined. Our model discriminates between two different kinds of combining the contributed values, which finally lead to a classification of rules in two categories.

The obvious way of combining contributed values is to take all of them into account. Let us assume that the application of all rules to a given document results in the sequence $\langle a_1, a_2, \ldots, a_m \rangle$, where each a_i is a contributed value of interest and an element of the set $\mathcal{E}$. Then, the formula for the calculation of the total score is the following:

$$\textit{score} \quad = \quad \| \, a_1 \oplus a_2 \oplus \ldots \oplus a_m \, \|$$

Figure 2: Properties of operators used in the combination of values of interest.

$a \oplus (b \oplus c) = (a \oplus b) \oplus c$	$\parallel a \parallel = \parallel b \parallel \Rightarrow \parallel a \oplus b \parallel = \parallel a \parallel = \parallel b \parallel$
$a \oplus b = b \oplus a$	$\parallel a \oplus \overline{a} \parallel = 0$
$a \oplus \mathbf{O} = a$	$\overline{\mathbf{O}} = \mathbf{O}$
$-1 \leq \parallel a \parallel \leq +1$	$\overline{a \oplus b} = \overline{a} \oplus \overline{b}$
$\parallel \mathbf{O} \parallel = 0$	$\parallel \overline{a} \parallel = - \parallel a \parallel$
$\parallel a_1 \parallel \leq \parallel a_2 \parallel \Rightarrow \parallel a_1 \oplus b \parallel \leq \parallel a_2 \oplus b \parallel$	

In this formula, operator $\oplus$ represents the combination of two values of interest, resulting in a third value, whereas operator $\parallel \cdot \parallel$ translates a value of interest (i.e. an element of $\mathcal{E}$) to a value in the interval $[-1, +1]$ of real numbers. The presence of an operator that negates values of interest is also necessary. This operator will map value a to its *negative* value, denoted by $\overline{a}$. The choice of operators $\oplus$ and $\parallel \cdot \parallel$, as well as of the negation operator must be made in an appropriate way, so that some properties are satisfied. The desired properties that these operators must have are shown in Fig. 2.

An obvious choice for the definition of set $\mathcal{E}$ is the interval $[-1, +1]$ of real numbers, which renders the presence of operator $\parallel \cdot \parallel$ unnecessary. Unfortunately, this choice makes it impossible to define the other operators in such a way as to satisfy all the desired properties.[1] For this reason, $\mathcal{E}$ must be defined in a different way. Each element a of $\mathcal{E}$ is taken to be a pair of the form $\langle x, n \rangle$, where x is a real number in the interval $[-1, +1]$ and n is a natural number. The value of x represents a value of interest. The value of n expresses the *multiplicity* of the value, i.e. the weight that this value carries. It is easy to confirm that the operators defined as follows satisfy the desired properties.

$$\begin{aligned}
\langle x_1, n_1 \rangle \oplus \langle x_2, n_2 \rangle &= \begin{cases} \left\langle \dfrac{n_1 x_1 + n_2 x_2}{n_1 + n_2}, n_1 + n_2 \right\rangle & \text{, if } n_1 + n_2 \neq 0 \\ \langle 0, 0 \rangle & \text{, if } n_1 + n_2 = 0 \end{cases} \\
\parallel \langle x, n \rangle \parallel &= x \\
\overline{\langle x, n \rangle} &= \langle -x, n \rangle \\
\mathbf{O} &= \langle 0, 0 \rangle
\end{aligned}$$

In other words, operator $\oplus$ is the weighted mean of the two values of interest.

It is sometimes useful to have certain values of interest that prevail over others. We can thus classify values in two categories: the "privileged" ones and the "common" ones. If the sequence of contributed values for a given document contains only common values, the formula that was given above can still be used. The same is true if only privileged values are present. However, if the sequence contains values from both categories, the total score is computed based only on the privileged values using the same formula; common values are ignored. An example of using such privileged values of interest is a hypothetical user's request to always reject documents written by a certain author, e.g. because the

[1]This choice makes it necessary to define $\overline{a}$ as $-a$, and then the associativity property for $\oplus$ cannot be satisfied, if one takes $a = b = +1$ and $c = -1$.

Figure 3: Algorithm for content-based filtering.

```
for each header $H$ in the area of arrival do
    score := unspecified
    for each rule $K$ in the base of authoritative rules do
        if the condition of $K$ is true for $H$ then
            score := $\max\{$ score, interest value of $K$ $\}$
    if the score is still unspecified then
        score := O (neutral)
        for each rule $K$ in the base of additive rules do
            if the condition of $K$ is true for $H$ then
                score := score $\oplus$ interest value of $K$
    if $\|$ score $\| >$ threshold for acceptance then
        place the document in the area of accepted documents
    else if $\|$ score $\| <$ threshold for rejection then
        place the document in the area of rejected documents
    else
        place the document in the area of neutral documents
```

author is not reliable, even if there are many other reasons to accept them.

It would be rather difficult to include this requirement in the mathematical model for values of interest. For this reason, it is not the values that are classified as privileged or common, but the rules that introduce them are classified in two categories: *additive* rules, which introduce common values, and *authoritative* rules, which introduce privileged values. In order to make this distinction clear, in our model we store the rules in two separate knowledge bases. The complete algorithm for the score calculation that is used in our model is shown in Fig. 3.

3.5. FILTERING CONTROL

The process of filtering control is performed by a specialized agent that interacts heavily with the user. After all, the user is the only one who can finally decide whether a document is interesting or not. For this reason, the user is called to check the decisions made by the filtering agent and to classify the documents that were placed in the three aforementioned areas in two new areas: the area of *finally accepted documents* and the area of *finally rejected documents*. With the new classification every combination of areas for a document transitions is possible. As the knowledge base is enriched, it is expected that transitions between areas of differing degree of interest will not be frequent. In the ideal case, the area of rejected documents will only contain uninteresting documents and the user will not have to check it at all, the area of accepted documents will only contain documents that the user will eventually find interesting, and the neutral area will contain as few documents as possible.

Filtering control is divided in two parallel processes: The *training process* aims at the update and enrichment of the knowledge base with the filtering criteria that the user actually applies. In this phase, the user classifies the documents that the agent has placed in the neutral area, unable to correctly classify them. The user has the option to indirectly indicate the criteria according to which he makes the classification by means of a friendly user interface. In this way, the user interacts naturally with the knowledge base by adding or updating rules, without knowing the form and the coding in which these rules exist in the knowledge base. On the other hand, the *error correction process* aims at the correction of errors that were made by the agent in the classification of documents. During this phase, the user classifies the documents that the agent has placed in the other two areas. This process does not affect the knowledge base, but only the filtering agent's degree of confidence.

The process of filtering control may result in changes in the system's state, that is, the knowledge base and the degree of confidence. Such changes will affect the outcome of future document classification. Implementations may provide the option of re-evaluating documents that have already been classified, according to the new state. Apart from that, our model does not address the issue of consistency between the current classification and the current state. In general, classification of a document is bound to be consistent with some state that has existed in the past, but not necessarily the current one.

3.6. INFORMATION SEEKING

The need for content-based document filtering would not be as imperative if the stream of information contained interesting documents with a high probability. This probability can be increased if the users are able to indicate to the system the sources that generate information that is interesting to them. Subsequently, the system must be able to direct information from these sources to the stream of information that reaches the users.

This process is performed by the *information seeking agent*. This agent interacts with the user and, by means of examples, observation or direct guidance, locates the information sources that the user finds interesting and reliable. Then, the same agent undertakes the collection of information from these sources in frequent or prearranged time intervals. The information seeking agent is not a part of the document filtering system. However, its general description is contained in the model because the problem that it tries to solve is very closely related to that of document filtering.

3.7. DISTRIBUTED DOCUMENTS

Efficient access of documents that are distributed over an intranet or the Internet is a key issue in all realistic implementations of our model. The client-server architecture has been successfully used in similar applications and is therefore considered as the most appropriate for implementing our model. In such implementations, servers would typically be responsible for the collection of documents, header construction and information seeking, whereas clients would request and filter document headers, request documents upon user's guidance and contribute to the information seeking process.

With all this in mind, our model in Fig. 1 can be divided in two parts: the client part, contained in the dotted rectangle, and the server part. In a multi-user implementation of our model, the filtering agent and the knowledge base could be part of a second kind of server.

4. Implementation

The suggested model has been used for the partial implementation of an experimental system for electronic document filtering in the environment of a journalist's office. The system, which is named ALEC, is hosted over a local area network of personal computers running Microsoft Windows with direct connection to the Internet and has been developed using Microsoft Visual C++. Apart from the aforementioned simplifications (A1) and (A2), the following limitations were also imposed on ALEC's implementation:

(A3) The header constructing agent can only manage documents of a given form.
(A4) The types of conditions supported are the simplest possible.
(A5) The information seeking agent has not been implemented.

ALEC's capabilities are still limited and the process of updating the selection criteria during filtering control is not yet as indirect as it is desired. The process of header construction is still premature.

A snapshot from ALEC's use is shown in Fig. 4, where, in the process of filtering control, ALEC shows the user a document that he could not classify. The document received a total score of 0.20. From the list of fields, the user can see that an interest value of 0.70 was attributed to the document because its subject was music (apparently ALEC has decided that the user is interested in music) and a negative value of -0.30 because the artist that it deals with is Johann Sebastian Bach (which ALEC considers not one of the user's favourites). The user can direct ALEC to change his selection criteria through this dialog box.

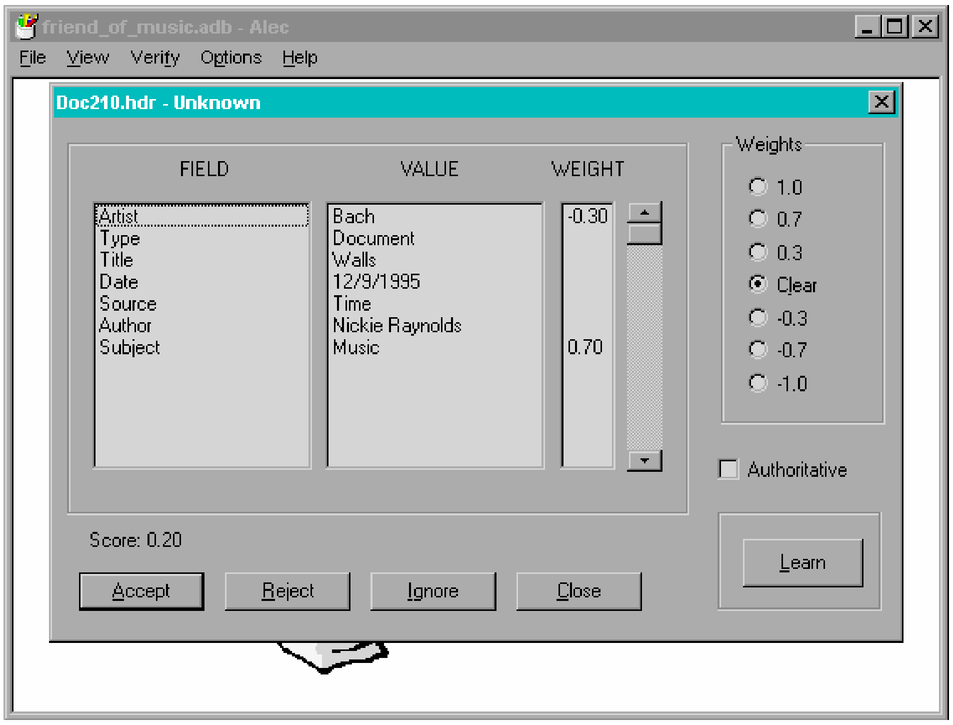
The evaluation of the experimental implementation is not yet complete and there are no trustworthy and adequate in volume performance results from its use. As far as this implementation is concerned, future research will aim at its improvement, on the one hand, and on its complete evaluation in a real environment, on the other.

5. Future directions

The primary target of our future research is the evaluation of our model with the development of realistic implementations. Useful results for the future improvement of our model may also stem from research in the field of *full-text retrieval*. If the filtering agent is capable of searching the full text of the electronic documents, it is possible to specify conditions that are based in the presence and the relative position of words in the text. The technology of full-text retrieval applications is already widely used by various information-seeking agents in the World Wide Web, with very encouraging results. However, as far as our filtering model is concerned, we do not see fit to abolish the stage of header construction.

In order to overcome limitation (A1), additional work is necessary, aiming at the support of multiple knowledge bases that will either belong to different users, or represent different interests of the same user. A possible direction that will be considered is that of integrating all knowledge bases in a global knowledge base, which may also contain generally accepted selection criteria, as well as common characteristics of the system.

Figure 4: Snapshot from ALEC’s dialog for filtering control.



Doc210.hdr - Unknown

FIELD	VALUE	WEIGHT
Artist	Bach	-0.30
Type	Document	
Title	Walls	
Date	12/9/1995	
Source	Time	
Author	Nickie Raynolds	
Subject	Music	0.70

Weights: ○ 1.0 ○ 0.7 ○ 0.3 [✓ Clear] ○ -0.3 ○ -0.7 ○ -1.0

☐ Authoritative

Score: 0.20

Finally, to overcome limitation (A2), further research will establish an interaction between the header construction agent and the knowledge base, allowing the construction of headers to reflect the user needs. The possibility of creating personalized headers for each user, according to the their individual selection criteria that are contained in the knowledge base, will also be examined.

6. Conclusion

In this chapter we propose a model for automating the tedious and time-consuming process of content-based electronic document filtering. The model is based on a set of collaborating intelligent agents, which mediate between the sources and the targets of information. Emphasis is given the homogeneous treatment of electronic documents, personalization, automatic training, dynamic adaptation to the user needs and believability.

Despite the fact that an implementation of the suggested model is still in a premature stage and that no substantial and trustworthy results from its use are yet available, the first experiences from its application to a journalist’s office are encouraging. We believe that, in the future, intelligent agent technology will provide adequate solutions to a lot of problems concerning the automation of tasks that currently make us less productive, including the content-based filtering of electronic documents.

Acknowledgements

This research work is funded by the Greek General Secretariat of Research and Technology, as part of the Programme for Supporting Research (ΠΕΝΕΔ 95) — Project #463.

References

[1] A. Reinhardt. Managing the new document. *BYTE*, pages 91–104, August 1994.

[2] P. Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):31–40, 146, July 1994.

[3] M. R. Genesereth. An agent-based approach to software interoperability. In *Proceedings of the DARPA Software Technology Conference*, 1992.

[4] Y. Lashkari, M. Metral, and P. Maes. *Collaborative Interface Agents*. MIT Press, Cambridge, MA, 1994.

[5] B. Sheth and P. Maes. Evolving agents for personalized information filtering. In *Proceedings of the Ninth Conference on Artificial Intelligence for Applications*, pages 345–352. IEEE Computer Society Press, 1993.

[6] R. Kozierok and P. Maes. A learning interface agent for scheduling meetings. In *Proceedings of ACM SIGCHI International Workshop on Intelligent User Interfaces*, pages 81–88, New York, NY, 1993. ACM Press.

[7] T. Selker. Coach: A teaching agent that learns. *Communications of the ACM*, 37(7):92–99, July 1994.

[8] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92, 1993.

[9] The Dublin Core Metadata Element Set. URL: http://purl.org/metadata/dublin_core/.

[10] The Rufus System. URL: http://www.almaden.ibm.com/cs/showtell/rufus/.

[11] K. Shoens, A. Luniewski, P. Schwarz, J. Stamos, and J. Thomas. The Rufus system: information organization for semi-structured data. In *Proceedings of the 19th Conference on Very Large Databases (VLDB '93)*, pages 97–107, Dublin, Ireland, 1993.

[12] M. Tresch, N. Palmer, and A. Luniewski. Type classification of semi-structured documents. In *Proceedings of the 21st Conference on Very Large Databases (VLDB '95)*, pages 263–274, Zurich, Switzerland, 1995.

[13] D. Smith and M. Lopez. Information extraction for semi-structured documents. In *Proceedings of the Workshop on Management of Semistructured Data, in conjunction with PODS/SIGMOD*, Tucson, AZ, USA, May 1997.