Thesis Proposal

PERSONALIZED AGENTS BASED ON CASE-BASED REASONING AND TRUST IN THE COLLABORATIVE WORLD

Miquel Montaner

Universitat de Girona
Av. Lluís Santaló s/n
17071 Girona
Catalonia (Spain)

http://eia.udg.es/~mmontane
miquel.montaner@udg.es

Director: Josep Lluís de la Rosa

July, 2001

Departament d'Electrònica, Informàtica i Automàtica
Universitat de Girona

Contents

List of Figures

List of Tables

List of Equations

1. PREFACE

1.1. Motivation

The introduction of Internet, World Wide Web, communications networks, and widespread computation and storage capabilities, has resulted in a global information society with growing users around the world. Information, the precious raw material of the digital age, has never been so easy to obtain, process and disseminate through the Internet. Yet, with the avalanche of information at our doors, there is a rapidly increasing difficulty of finding what we want, when we need it, and in a way that better satisfies our requirements.

Users are constantly confronted with situations in which they have many options to choose from and need assistance exploring or winnowing down the possibilities. Internet Search Engines commonly find thousands of potentially relevant sites. In applications, a user is required to specify his information need in terms of a query which is then compared (typically at a simple keyword level) with documents in a collection and those likely to be most related to the query and thus potentially relevant to the user.

Recently, in the Artificial Intelligence community, there has been a great deal of work on how AI can help to solve this problem. Notions of personalized search engines, intelligent software agents, and recommender systems have gained large acceptance among users for the task of assisting them in searching, sorting, classifying, filtering and sharing the vast amount of information now available on the Web. The combination of the modeling of preferences of particular users, building content models, and the modeling of social patterns in intelligent agents [Maes, 1994] would provide users with means for managing information in a rational way, and, thus, helping to overcome the information overload.

As software agents act in the information space (virtual), physical agents execute in the real world. Research on both kinds of agents has been performed separately. However, we think that most of the results obtained in the physical world can be applied to the software space. For instance, taking advantage of the knowledge about the physical body obtained by introspection that has been studied for several years at our university: the University of Girona (UdG) [de la Rosa et al., 1999], [Oller et al., 2000]. The same way physical agents make distinction among robots, because robots are all physically different, the physical agents potentially can make distinction among human beings, because people are all “different”. Since physical agents commit to better actions according to introspected capabilities from the physical body, the same way physical agents would commit to better transactions according to introspected tastes from the human beings. Thus, automatic control oriented concepts as excitation, dynamics, and stability of the physical bodies can be imported to the personalization field.

How can be performed this real-to-virtual mapping of properties and behaviors is the main goal of our work. We expect to obtain a general concept of personalization agents that improves the state of the art in terms of performance and maintainability. Inversely, new features about the personalization agents could be also mapped to the physical agents discipline.

We constraint our research on the implementation of agents with CBR techniques. Accordingly to our experience (i.e., [Meléndez et al., 2000] and [Meléndez et al., 2001]), CBR has excellent applicability in the automatic control domain because of its maintainability, explicability, robustness and exception handling features, that makes the CBR globally better than any other knowledge representation scheme. Furthermore, it is very easy to create hybrid evolutionary and

CBR algorithms that our physical agents need for some sort of collaborative personalization. However, CBR has still drawbacks (no perfect technique has been developed, yet), among others, the decision about frequently past non-useful experiences. Our aim is also to cope with this problem and develop a mechanism to control obsolete cases. This will be a contribution to the CBR research.

1.2. The Problem

To test our development, we are thinking on an application in the e-commerce field, presumably a personal restaurant recommendation agent. It is a quite constrained domain that offers a wide range of personal tasks in order to build several patterns of personalization. Our aim is to work with a real application requirement also inherited from the physical agents experience. The UdG is working on a European project to design an intelligent drying chamber for the curing of hams and sausages [Euroagri-Indrycha], which can be a good starting point.

1.3. Contributions

The main contribution of this thesis is the application of two techniques to personalization: CBR and trust in multi-agent systems. CBR is able to utilize knowledge of previously evaluated items for the user to guess the interest of the user about new items. The CBR nature provides user acceptance and easy maintenance to the system. Trust in multi-agent systems is a technology that allows agents thinking about the others as personal entities in which you can rely on or not. The proposal of trust in the collaborative world makes personal agents more robust in front of the other agents.

A preliminary contribution in order to reach our main goal is to build a state of the art about personalized agents on the Internet. The state of the art is organized as a taxonomy, which classifies personalized systems into 10 general features. This taxonomy could be a starting point to researchers that want to build a personalized agent.

1.4. Outline of the Thesis Proposal

The thesis proposal has been organized in several sections.

Section 2, State of the Art: Several areas related to this work are surveyed. The state of the art is organized as a taxonomy to classify the current personalized systems on the Internet.

Section 3, The Proposal: Built on the state of the art, the goals of this work are outlined.

Section 4, Preliminary Work: a preliminary example has been developed to exhibit the thesis proposal viability.

Section 5, Planning: a work planning is presented to achieve the expected results in the next two years.

Section 6, Expected Results and Further Work: Expected results, future improvements and new research areas are introduced.

2. STATE OF THE ART

2.1. Introduction

Some papers present a state of the art about personalized systems (e.g., [Sarwar et al., 2000], [Pretschner and Gauch, 1999], [Terveen and Hill, 2001], [Kobsa et al., 2001]). [Schafer et al., 2001] present a taxonomy of recommender systems in the field e-commerce but only classify the used techniques into three features. This paper presents a more complete taxonomy of general intelligent personalized agents on the Internet based on the current state of the art. 37 different systems from different domains are studied (i.e., web recommenders, personalized newspapers, movie recommenders or e-mail filtering).

NAME	REFERENCES	DOMAIN
ACR News	[Mobasher et al., 2000]	Netnews Filtering
Amazon	[Amazon]	Comerc electronic
Amalthaea	[Moukas, 1997]	Web Recommender
Anatagonomy	[Sakagami et al., 1997]	Personalized Newspaper
Beehive	[Huberman and Kaminsky, 1996]	Sharing News
Bellcore Video Recommender	[Hill et al., 1995]	Movie Recommender
Casmir	[Berney and Ferneley, 1999]	Document Recommender
CDNow	[CDNow]	Comerc electronic
Fab	[Balabanovic and Shoham, 1997]	Web Recommender
GroupLens	[Resnick et al., 1994]	Netnews Recommender
ifWeb	[Minio and Tasso, 1996], [Asnicar and Tasso, 1997]	Web Recommender
InfoFinder	[Krulwich and Burkey, 1995], [Krulwich and Burkey, 1996]	Information Recommender
INFOrmer	[Riordan and Sorensen, 1995], [Sorensen et al., 1997]	Netnews Filtering
Krakatoa Chronicle	[Kamba et al., 1995]	Personalized Newspaper
LaboUr	[Schwab et al., 2001]	Document Recommender
Let's Browse	[Lieberman et al., 1999]	Web Recommender
Letizia	[Lieberman, 1995]	Web Recommender
LifeStyle Finder	[Krulwich, 1997]	Purchase, Travel and Store Recommender
MovieLens	[Good et al., 1999]	Movie Recommender
News Dude	[Billsus and Pazzani, 1999]	Netnews Recommender
NewsWeeder	[Lang, 1995]	Netnews Recommender
NewT	[Sheth and Maes, 1993]	Netnews Filtering
Personal WebWatcher	[Mladenic, 1996]	Web Recommender
PSUN	[Sorensen and McElligot, 1995]	Netnews Recommender
Re:Agent	[Boone, 1998]	E-mail Filtering
Recommender	[Basu et al, 1998]	Movie Recommender
Ringo / FireFly	[Shardanand and Maes, 1995], [Shardanand, 1994]	Music Recommender
SIFT Netnews	[Yan and Garcia-Molina, 1995]	Netnews Filtering
SiteIF	[Stefani, and Strappavara, 1998]	Web Recommender
Smart Radio	[Hayes and Cunningham, 1999], [Hayes and Cunningham, 2000]	Music Lists Recommender
Syskill & Webert	[Pazzani et al., 1996], [Pazzani and Billsus, 1997]	Web Recommender
Tapestry	[Goldberg et al., 1992]	E-mail Filtering
Webmate	[Chen and Sycara, 1998]	Web Recommender
WebSail	[Chen et al., 2000]	Web Search Filtering
WebSell	[Cunningham et al., 2001]	Purchase Recommender
Websift	[Cooley et al., 1999]	Web Recommender
WebWatcher	[Armstrong et al., 1995], [Joachims et al., 1997]	Web Recommender

Table 1. Domain of the Analyzed Systems

Table 1 shows the domain of the different analyzed systems. These systems and their references are deeply analyzed to extract a set of 10 common features. The 10 features are used to classify the personalized agent, thus providing a taxonomy of the systems.

This paper is organized as follows. First, we present the 10 features that constitute the taxonomy. Then, we proceed through section 3 to 12 by providing the classification of the systems according to each feature. We end at section 13 with several conclusions.

2.2. A Taxonomy of Personalized Agents on the Internet

The process of filtering Web documents, separating relevant documents from non-relevant ones, or recommending items such as CDs, books movies, etc., can be viewed as a personalized task based on user profiles, which are somewhat hypothesis of unknown target concepts of user preferences. Intelligent agents build and exploit these profiles. The analysis of 37 personalized systems has result in the identification of 10 common features of generation and exploitation of user profiles. These features establish a taxonomy under which the different systems can be classified.

The purpose of this section is to explain the taxonomy features. First, we will discuss the features that characterize profile generation and maintenance. Second, we proceed by outlining the features regarding user profile exploitation. Then, we explain the two last features related to general aspects as system evaluation and architecture. And we end by summarizing the 10 features.

2.2.1. Profile Generation and Maintenance

A user profile is a representation of the user tastes, interests and/or preferences, and it is the basic feature of a personalized system. To generate and maintain this profile we notice five design decisions that constitutes the first five features of our taxonomy: the profile representation technique, the technique to generate the initial profile, the source of the relevance feedback that represents the user interests, the profile learning technique and the profile adaptation technique. Figure 1 shows the relation of these techniques in the generation and maintenance of user profiles.

The *profile representation* is the first step to take into account in a personalized system, since the other techniques depend on it. Once this step is decided, the other techniques can be defined. A personalized system cannot start its function until the user profile is created, and, moreover, it is desirable to know as much as possible from the user so that the systems provide satisfactory results to the user from the very beginning. Therefore, systems must use a suitable technique to *generate* an accurate *initial profile*.

To generate and maintain the user profile, the system needs relevant information about the user's interests. When users interact with a computer, they provide a great deal of information about themselves. Successful interpretation of these data streams is necessary for computers to tailor themselves to each individual's behavior, habits and knowledge. Our computers support many different applications, each of which does one thing well: showing users mail, providing them with an electronic datebook, letting them play a game. As from the interaction of the user with these applications, the system can gather *relevance feedback* to know his tastes, interests or preferences. Typically, the feedback given explicitly or implicitly by the user has no sense itself. Therefore, there is a need of a *profile learning technique* that extracts the relevant information and structures this information depending on the representation of the profile.

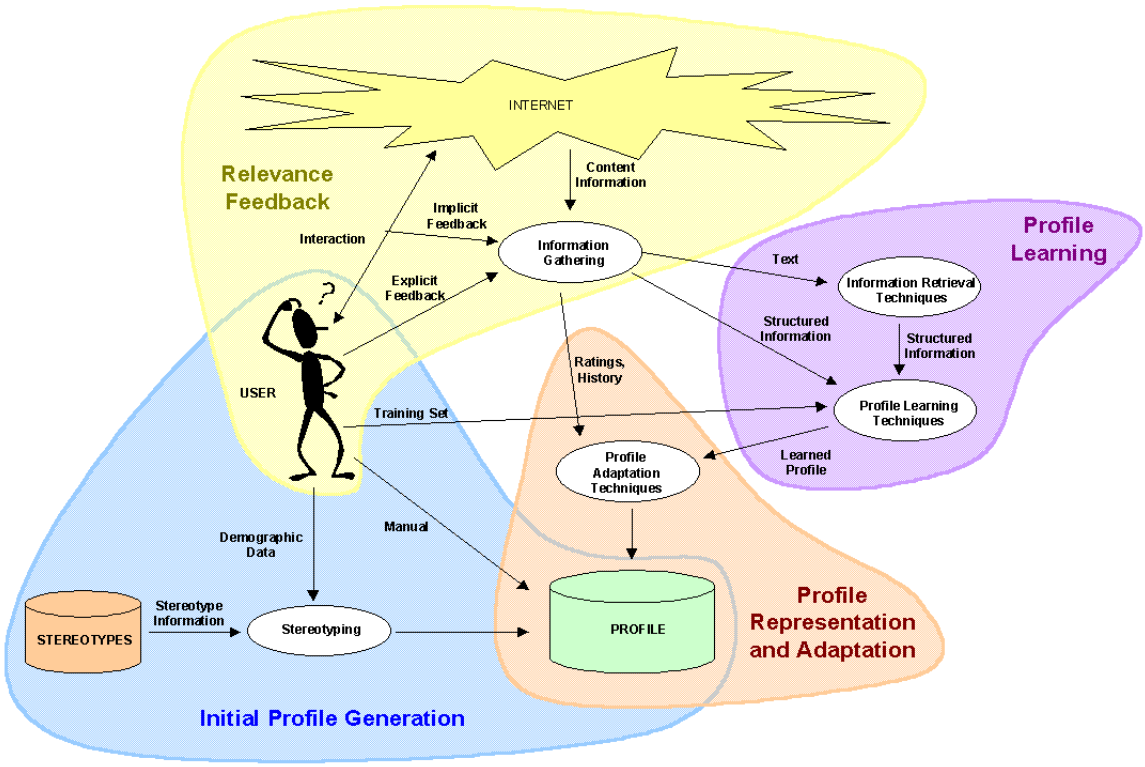

Figure 1. Profile Generation and Maintenance

The user profile is used to filter information. User tastes usually change as time proceeds. So, the user profile should also be changed in order to retain the desired accuracy in filtering. That is, human interests change with time and there is a need of a *technique to adapt the user profile* to the new interests and to forget the old ones.

2.2.2. Profile Exploitation

Once the user profile is created, the systems exploit it, for example, to filter incoming Netnews, to recommend interesting restaurants,... This paper is focussed on recommendation systems [Sangüesa and Cortés, 2000]. We think that other functionalities can be viewed as a kind of recommendation. For instance, systems that proactively filter e-mail messages can be viewed as a system that recommends actions for the new messages and execute them when the confidence is very high.

Recommendation systems made decisions according to the information available. Since there is so much information on the web, a fundamental issue on such systems is to select the adequate information upon which perform the decisions. That is to say, the need of an *information filtering method* is essential in recommender system. There are three main information filtering methods: demographic, content-based and collaborative. Demographically, similar people tend to behave in a similar way. Demographic filtering systems use the general features of a cluster of similar people or a stereotype of a person to infer the interests of a particular user. Content-based filtering approaches recommend items for the user based on the descriptions of the previous evaluated items, in other words, they recommend items because they are similar to

items the user has liked in the past. Several *user profile-item matching methods* can be used in order to compare the representation of the user interest and new items. But when content-based applications can make use of a common database of information about the user, and communicate with one another about the user, their ability to personalize themselves increases dramatically. Collaborative Filtering systems matches people with similar interests and then makes recommendations on this basis. Different *methods* are used by the systems to *match user profiles* and find users with similar interests.

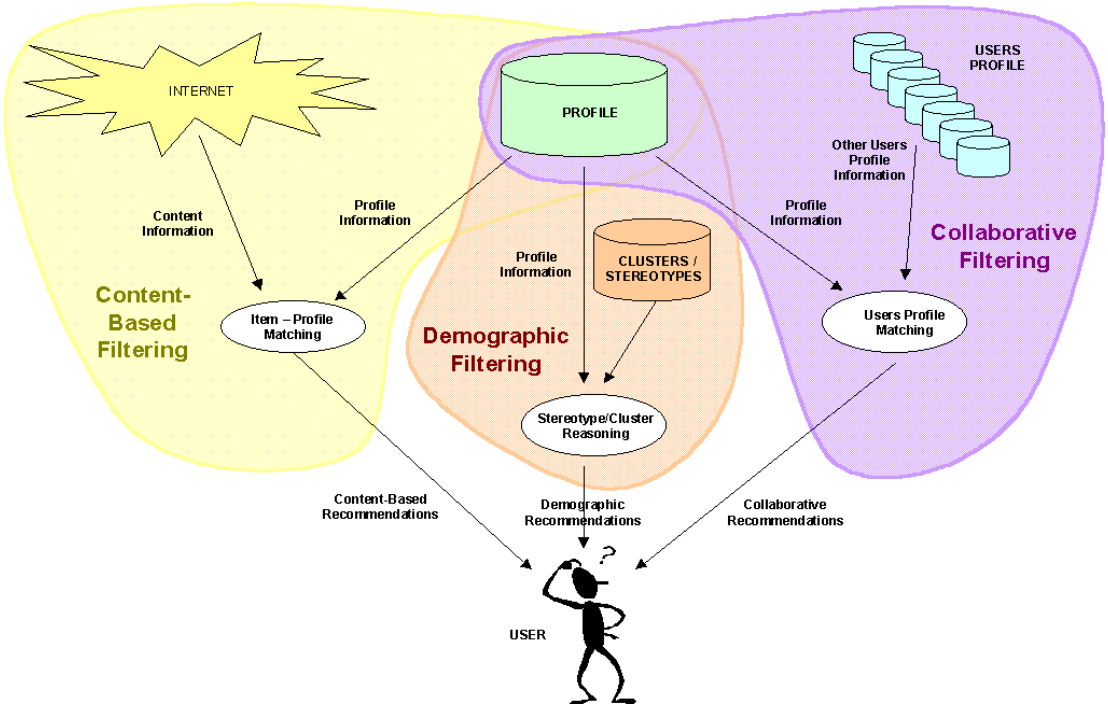

Figure 2. Profile Exploitation for Recommendation

Regarding exploitation, we distinguish, then, three main features: the information filtering method, the item-profile matching and the user profile matching techniques.

2.2.3. Other Issues

Further of user profile generation and maintenance, developers have to take into account other issues as the architecture and evaluation of the system. The general *architecture of the system* conditions the whole development, thus, it is an important feature to take into account. For simplicity purposes, in the whole paper, the general word "system" is used to mention the current personalized applications. However, some applications are structured as intelligent agents or ecosystems of agents.

Regarding evaluation, unfortunately, only a few systems evaluate and discuss their results scientifically. This is in part due to the fact that it actually is hard to determine how well a personalization systems works, as this involves purely subjective assessments. However, most of the analyzed systems present results based on different *evaluation methods*.

The last two fields that form the taxonomy are the system architecture and evaluation.

2.2.4. The Ten Classification Features

Summarizing, the ten features of the taxonomy are the following:

Profile Generation and Maintenance:

- User Profile Representation
- Initial Profile Generation
- Relevance Feedback
- Profile Learning
- User Profile Adaptation

Profile Exploitation:

- Information Filtering Method
- User Profile – Item Matching Techniques
- User Profile Matching Techniques

Other Issues:

- System Architecture
- Evaluation of the System

In the following sections, the different features are deeply analyzed and the techniques used in the state of the art to implement it are exposed.

2.3. Information Filtering Method

Early, Malone et al. propose three types of information filtering activities: cognitive, economic and social [Malone et al., 1987]. Cognitive activities filter information based on content. Economic filtering activities filter information based on estimated search cost and benefits of its use. Social activities filter information based on individual judgments of quality communicated through personal relationships.

These three information filtering activities proposed by Malone et al. have evolved, mainly, in three information filtering approaches for making recommendations: demographic filtering, content-based filtering and collaborative filtering. Demographic filtering approaches use descriptions of the people to learn a relationship between a single item and the type of people that like that object. This is a new approach emerged from the stereotypes proposed by Rich [Rich, 1979]. Content-based filtering approaches use descriptions of the content of the items to learn a relationship between a single user and the description of the items. That is the evolution of the cognitive activities. Collaborative filtering approaches use the feedback of a set of people on a set of items to make recommendations, but ignore the content of the items or the descriptions of the people. This is the evolution of the social activities. However, the economic activities have not yet been implemented.

Table 2 shows the information filtering techniques used by the different analyzed systems.

NAME	METHOD
ACR News	Content-Based Filtering
Amazon	Hybrid
Amalthaea	Content-Based Filtering
Anatagonomy	Hybrid
Beehive	Collaborative Filtering
Bellcore Video Recommender	Collaborative Filtering
Casmir	Hybrid
CDNow	Hybrid
Fab	Hybrid
GroupLens	Collaborative Filtering
ifWeb	Content-Based Filtering
InfoFinder	Content-Based Filtering
INFOrmer	Content-Based Filtering
Krakatoa Chronicle	Hybrid
LaboUr	Hybrid
Let's Browse	Content-Based Filtering
Letizia	Content-Based Filtering
LifeStyle Finder	Demographic Filtering
MovieLens	Hybrid
News Dude	Content-Based Filtering
NewsWeeder	Hybrid
NewT	Content-Based Filtering
Personal WebWatcher	Hybrid
PSUN	Content-Based Filtering
Re:Agent	Content-Based Filtering
Recommender	Hybrid
Ringo / FireFly	Collaborative Filtering
SIFT Netnews	Content-Based Filtering
SiteIF	Content-Based Filtering
Smart Radio	Collaborative Filtering
Syskill & Webert	Content-Based Filtering
Tapestry	Collaborative Filtering
Webmate	Content-Based Filtering
WebSail	Content-Based Filtering
WebSell	Hybrid
Websift	Hybrid
WebWatcher	Hybrid

Table 2. Information Filtering Method of the Systems

2.3.1. Demographic Filtering

Demographic filtering approaches use descriptions of the people to learn a relationship between a single item and the type of people that like that object. The user models are created by classifying users in stereotypical descriptions [Rich, 1979], representing the features of classes of users. Personal data about the user is required and is used to classify users in terms of these demographic data. Classifications are used as general characterizations for the users and their interests. Commonly, the personal data is asked to the user in a registration form (see section 2.5.3). The resulting profiles span the range of information contained in the demographic database.

For instance, the method implemented by Krulwich in the LifeStyle Finder [Krulwich, 1997] uses a commercially available database of demographic data that encompasses the interests of people nationwide. They have been using a demographic system called PRIZM from Claritas Corporation which divides the population of the United States into 62 demographic clusters according to their purchasing history, lifestyle characteristics and survey responses. The

demographic database contains information on more than 600 variables, each of which refers to a specific lifestyle characteristics, purchase or activity.

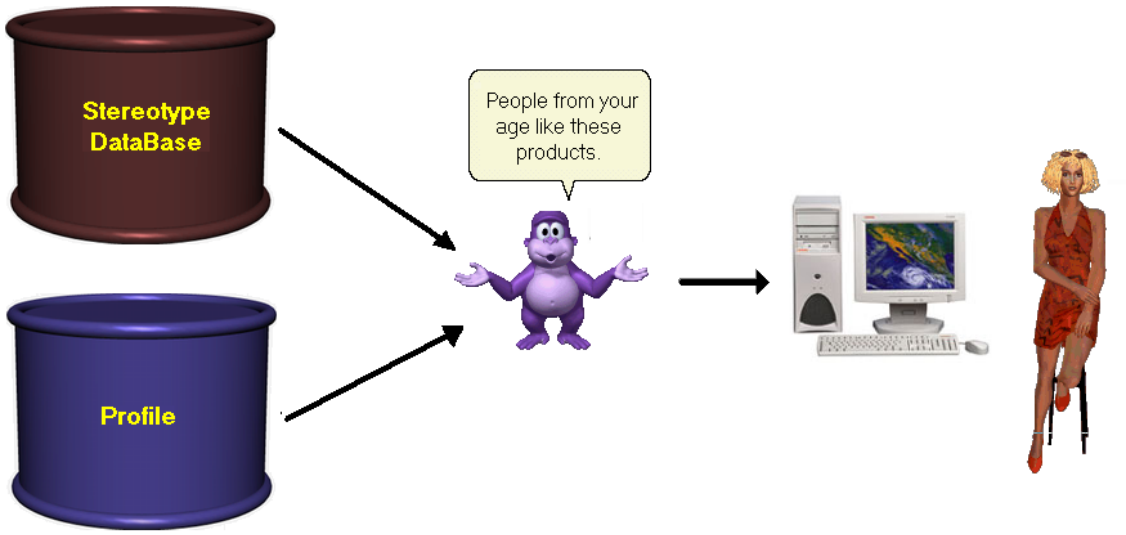

Figure 3. Demographic Filtering

However, a pure demographic filtering system has several shortcomings.

- Demographic filtering is based on a generalization of the user's interests, so the system recommends people with similar demographic profiles the same items. As every user is different the recommendations prove to be too general.

- The demographic approaches do not provide any individual adaptation to interest changes. The user's interests tend to shift over time [Koychev, 2000], so the user profile need to adapt through the time.

Nevertheless, demographic information can be a useful technique if combined with other approaches. This technique is very useful to generate the initial profile of the user (see section 2.5.3).

2.3.2. Content-Based Filtering

Content-based filtering approaches recommend items for the user based on the descriptions of the previous evaluated items, in other words, they recommend items because they are similar to items the user has liked in the past. User profiles are created using features extracted from the items (see section 2.7) and each user is assumed to operate independently.

The input data most often take the form of samples of the user's interests or preferences in a given area, and the profile is a generalization of these data that can be used generatively to carry out task on behalf of the user. A common application takes sample items that a user finds interesting or uninteresting and generates profiles of the user's interests. These profiles are then used to find or recognize other items that are likely to be of interest. Other common applications process input data such as movies or music albums, that the user likes and dislikes and use the resulting profiles to suggest new movies or albums to the user. Different methods are used by the systems to match a user profile with the new items and decide whether they are interesting for the user.

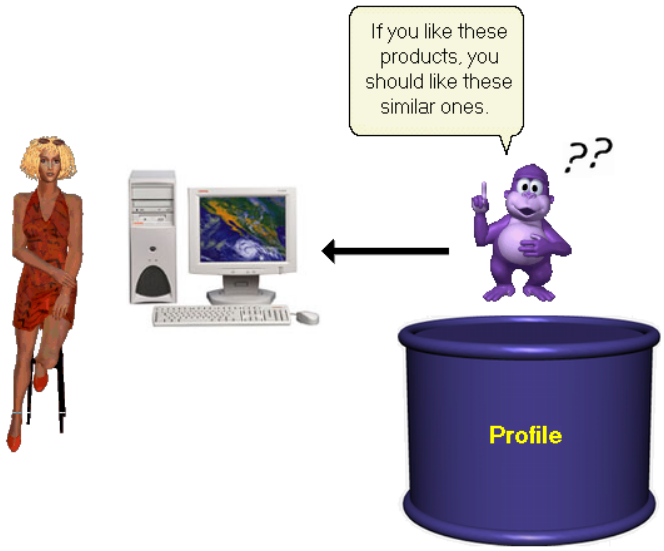

Figure 4. Content-Based Filtering

For instance, in Syskill&Webert [Pazzani et al., 1996] the user rates a number of Web documents from some content domain on a binary "hot" and "cold" scale. Based on these ratings, it computes the probabilities of words being in hot or cold documents. Lieberman developed the system Letizia [Lieberman, 1995], which assists a user in Web browsing. Letizia tries to anticipate interesting items on the Web that are related to the user's current navigation context. For a set of links Letizia computes a preference ranking based on a user profile. This profile is a list of weighted keywords, each one indicating the relevance of the words found on the pages. Personalized WebWatcher [Mladenic, 1996] observes individual user's choices of links on Web pages, in order to recommend links on other Web pages that it may visit later. The user does not have to provide explicit ratings. Instead, visited links are taken as positive examples, non-visited links as negative ones.

However, a pure content-based filtering system has several shortcomings.

- The content-based approaches are based on objective information about the items. This information is automatically extracted from some sources (e.g., Web pages) or introduced manually (e.g., product database). The selection of one item or another is mostly based on subjective attributes about the item (e.g., a well-written document or a product with a spice taste). Therefore, the attributes, which better influence the user choice, are not taken into account.

- Another problem, which has been studied extensively, is the over-specialization. Content-based filtering techniques have no inherent method for generating serendipitous finds. The system recommends more of what the user already has seen before (and indicated liking). When the system can only recommend items scoring highly against a user profile, the user is restricted to seeing items similar to those already rated. In practice, additional hacks are often added to introduce some element of serendipity like injecting a note of randomness-for example the crossover and mutation operations (as part of a genetic algorithm) have been proposed as a solution [Sheth and Maes, 1993].

- With the pure content-based approach, a user's own ratings are the only factor influencing future performance. However, only a few ratings are provided due to both the reluctance of

the users to perform actions that are not directed towards their immediate goals, if they do not receive immediate benefits [Carroll and Rosson, 1987], and the low interaction of the user with the system. Therefore, the recommendation quality has a low precision.

Nevertheless, these shortcomings can be solved combining the content-based approach with the collaborative filtering approach (see section 2.3.4).

2.3.3. Collaborative Filtering

The collaborative filtering technique matches people with similar interests and then makes recommendations on this basis. Recommendations are commonly extracted from the statistical analysis of patterns and analogies of data extracted explicitly from evaluations over items (ratings) given by the different users or implicitly by monitoring the behavior of the different users in the system. This approach is very different to the content-based filtering, the other most commonly used approach: rather than recommending items because they are similar to items a user has liked in the past, they are recommended based on other user's preferences. Rather than computing the similarity of the items, the similarity among users is computed. In this case a user profile consists simply of the data that the user has specified. This data is compared to those of others users to find overlaps in interests among users. These are used to recommend new items to the users. Typically, for each user a set of "nearest neighbors" is defined using the correlation between the past ratings. Scores for unseen items are predicted using a combination of the scores from the nearest neighbor. This approach requires less computation than the previous one because it doesn't have to reason about the user data, and it clearly leverages the commonalties between users.

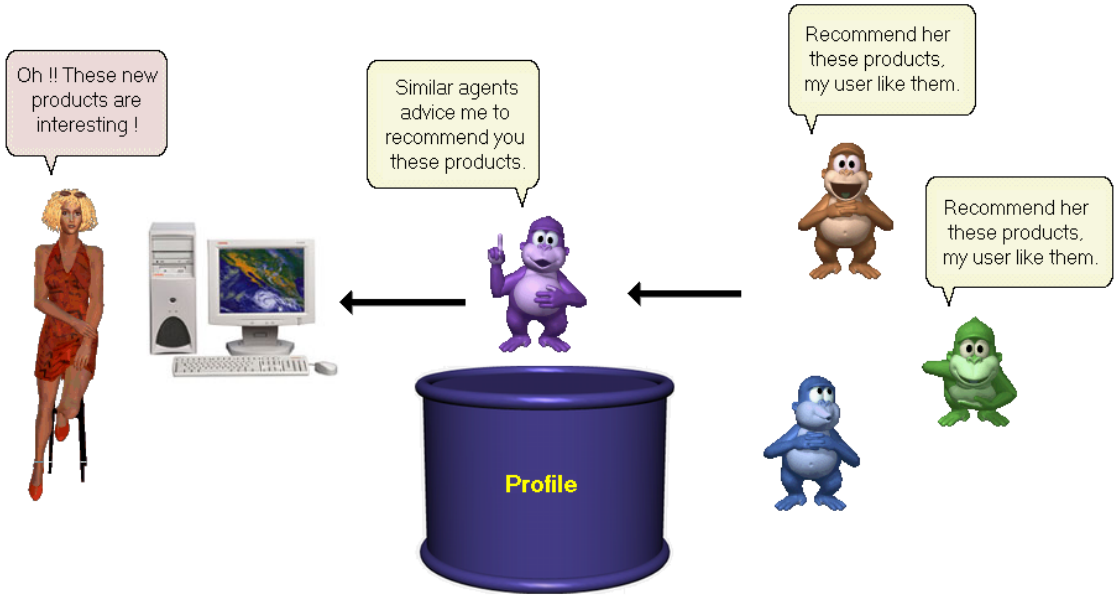

Figure 5. Collaborative Filtering

Terveen and Hill claim the necessity of three pillars to support this approach [Terveen and Hill, 2001]: many people must participate (increasing the likeliness that any person will find others users with similar preferences), there must be an easy way for representing user's interests in the system, and the algorithms must be able to match people with similar interests.

For instance, Tapestry [Goldberg et al., 1992] is one of the earliest implementations of collaborative filtering based recommender systems. This system relied on the explicit opinions of people from a close-knit community, such as an office workgroup. Another popular system is GroupLens [Konstan et al., 1997], which computes correlation between readers of Usenet newsgroups by comparing their ratings of news articles. The ratings of an individual user are used to find related users with similar ratings, and their ratings are processed to predict the user's interest in new articles.
However, a pure collaborative filtering system has several shortcomings:

- The early-rater problem: if a new item appears in the database there is no way it can be recommended to a user until some more information about it is obtained through another user either rating it or specifying which other items it is similar to. A collaborative filtering system provides little or no value when a user is the first one in his neighborhood to enter a rating for an item. In fact, the systems depend on the altruism of a set of users who are willing to rate many items without receiving many recommendations. Economists have speculated that even if rating required no effort at all, many users would choose to delay considering items to wait for their neighbors to provide them with recommendations. Without altruists, it might be necessary mechanisms to encourage early ratings.

- The sparsity problem: the goal of collaborative filtering systems is to help people focus on reading documents (or consuming items) of interests. Due to the last shortcoming, if the number of users is small relative to the volume of information in the system (because there is a very large or rapidly changing database) then there is a danger of the coverage of ratings becoming very sparse, thinning the collection of recommendable items. On he other hand sparsity poses a computational challenge as it becomes harder to find neighbors and harder to recommend items since few people have rated most of them.

- Another logic problem is that for a user whose tastes are unusual compared to the rest of the population there will not be any other users who are particularly similar, leading to poor recommendations.

- The difficulty of achieving a critical mass of participants makes collaborative filtering experiments expensive. Collaborative filtering systems require data from a large number of users before being effective, require a large amount of data from each user, and limit its recommendations to the exact items specified by the population of users. One clear disincentive in present experiments is the additional cognitive load imposed on the user by the requirement to provide explicit feedback.

- The critical dependency on the size and composition of the user population also influence a user's group of nearest neighbors. In a situation in which feedback fails to cause this group of nearest neighbors to change, expressing dislike for an item will not necessarily prevent the user from receiving similar items in the future. Furthermore, the lack of access to the content of the items prevents similar users from being matched unless they have rated the exact same items.

Nevertheless, these shortcomings can be solved combining the collaborative filtering approach with the content-based filtering approach (see section 2.3.4).

Herlocker et al. introduced also the problem of lack of transparency in the collaborative filtering systems [Herlocker et al., 2000]. Collaborative systems today are black boxes, computerized oracles that give advice, but cannot be questioned. A user is given no indicators to consult to determine when to trust a recommendation and when to doubt one. These problems have prevented acceptance of collaborative systems in all but the low-risk content domains. Therefore, the collaborative filtering systems are not trusted for high-risk content domains.

2.3.4. Hybrid

Hybrid systems exploit features of content-based and collaborative filtering, since they will almost certainly prove to be complementary. On the one hand, pure collaborative systems solve all of the shortcomings given for pure content-based systems. The first shortcoming of the content-based systems is the lack of subjective data about the items. In a collaborative system, the community of users can offer this kind of data explicitly. It can be considered like an opinion of the item that a confident friend offers you. For instance, you can buy a product because a user with similar tastes recommends you a spice product and you like spice products. Another shortcoming of the content-based systems is the lack of novelty. A perfect content-based technique would never find anything novel, limiting the range of applications for which it would be useful. Collaborative filtering techniques excel at identifying novelty using other user's recommendations and you can receive items dissimilar to those seen in the past. For instance, a user with similar tastes can recommend you products that you never have bought. Finally, content-based systems have the lack of user ratings to represent the user's interests. Collaborative systems can complete the user information with the other user's experience as a basis. For instance, if you are very similar to another user and you have not rated a product, you can use other user's ratings to complete the user's interests.

On the other hand, pure content-based systems solve all of the shortcomings given for pure collaborative systems. The first shortcoming of the collaborative systems is the early-rater problem. With the content-based methods, new items can be recommended on the basis of the content, without the necessity of explicit ratings. Content-based systems vanish the scarcity problem because the profile keep information about the content of the items, no the products with the ratings. With a content-based systems we can recommend a user with unusual tastes without the necessity of a similar user. Finally, the mass of participants is not important in content-based systems because they do not depend on the population.

Thus, both content-based and collaborative filtering contribute to the other's effectiveness, avoiding the limitations mentioned for each system and allowing an integrated system to achieve both reliability and serendipity. Several papers claim the outperform of the hybrid systems (e.g., [Pazzani, 1999] and [Good et al., 1999]).

Such systems as Fab [Balabanovic and Shoham, 1997], LaboUr [Schwab et al., 2001] or WebSell [Cunningham et al., 2001] propose a very simple method for combining the two approaches: user profiles based on content analysis are maintained, and these profiles are directly compared to determine users with similar preferences for collaborative recommendation. Users receive items both when they score highly against their own profile, and when they are rated highly by a user with similar profile. Using content-based recommendations can solve the problems of the unseen items by others. We can use the profile we build from the content of items to make recommendations to users even if there are no other users similar to them, and we can also filter out items similar to those disliked in the past. We can make collaborative recommendations even between users who have not rated any of the same items (as long as they have rated similar items), extending the reach of collaborative systems to include databases which change quickly or are very large with respect to the number of users. By utilizing group feedback we potentially require fewer cycles to achieve the same level of personalization.

2.4. User profile Representation

The construction of accurate profiles is a key task – the system's success will depend to a large extend on the ability to represent the user's actual interests. Accurate profiles enable both the

content-based component (to insure recommendations are appropriate) and the collaborative component (to insure users with similar profiles are indeed similar).

NAME	TECHNIQUE
ACR News	Frequent Itemsets, URL Clusters
Amazon	Purchase History with Ratings
Amalthaea	Weighted Feature Vector
Anatagonomy	Weighted Feature Vector
Beehive	Clusters (Weighted Feature Vector)
Bellcore Video Recommender	User-Item Ratings Matrix
Casmir	Weighted Feature and Document Network
CDNow	Purchase History with Ratings
Fab	Weighted Feature Vector
GroupLens	User-Item Ratings Matrix
ifWeb	Multivalued Weighted Attributes, Weighted Semantic Network
InfoFinder	Decision Tree
INFOrmer	Weighted Associative Network
Krakatoa Chronicle	Weighted Feature Vector
LaboUr	Probabilistic Feature Vector, Boolean Feature Vector
Let's Browse	Weighted Feature Vector
Letizia	Weighted feature vector
LifeStyle Finder	Demographic Features
MovieLens	Weighted Feature Vector, Inducted Rules
News Dude	Short Term: Weighted, Long Term: Probabilistic Feature Vector
NewsWeeder	Weighted Feature Vector
NewT	Weighted Feature Vector
Personal WebWatcher	Probabilistic Feature Vector
PSUN	Weighted n-grams
Re:Agent	Weighted Feature Vector, Neural Network
Recommender	Inducted Rules
Ringo / FireFly	User-Item Ratings Matrix
SIFT Netnews	Boolean Feature Vector, Weighted Feature Vector, Decision Tree
SiteIF	Weighted Semantic Network
Smart Radio	User-Item Ratings Matrix
Syskill & Webert	Probabilistic Feature Vector, Boolean Feature Vector, Decision Tree, Weighted Feature Vector
Tapestry	Indexed Messages and Annotations
Webmate	Weighted Feature Vector
WebSail	Boolean Feature Vector
WebSell	Interesting/Not Interesting Products
Websift	Inducted Rules, Patterns, Statistics
WebWatcher	Boolean Feature Vector

Table 3. Profile Representation Technique of the Systems

Several approaches of the user profile representations have been implemented: a history of purchases, web navigation or e-mails, an indexed vector of features, a n-gram, a semantic network, an associative network, a classifier including neural networks, decision trees, inducted rules or Bayesian networks, a matrix of ratings and a set of demographic features. Table 3 shows the user profile representation techniques used by the different analyzed systems.

2.4.1. History

Some systems keep as a user profile the list of purchases, the navigation history in WWW or the content of the e-mail boxes. Additionally it is also usual to keep the relevance feedback of the user associated with each item in the history. Systems based on history do not use any learning profile technique and they concentrate all the process in the profile explotation.

Such approach is most commonly used in e-commerce, where systems keep as a profile the list of purchased products and the user ratings, as relevance feedback. This is the case of the most popular state of the art personalized systems in e-commerce: Amazon.com [Amazon] and CDNow.com [CDNow]. A similar approach is used in WebSell [Cunningham et al., 2001], where the profile is defined using two lists, one with the purchased products rated as "interesting" and another with the "uninteresting" ones. Another approach is implemented in Tapestry [Goldberg et al., 1992], an e-mail filtering system that builds the profile while keeping track of the messages and annotations given by the user.

2.4.2. Vector Space Model

The items are represented with a vector of features, usually words or concepts, with a value associated. This value can be a Boolean or a real number. The Boolean value represents the presence of the value of the feature, and the real number represents the frequency, relevance or probability of the feature calculated with information indexing (see section 2.7.2.2).

2.4.2.1. Binary / Boolean Vector Space Model

The items are represented with a vector of features with a Boolean value. This value typically represents whether the feature is present in the item or not.

2.4.2.2. Weighted Vector Space Model

The items are represented with a vector of features with a weight (a real number). The data that are potentially available for calculating the weight are the frequency of occurrence of the processing token in an existing item (i.e., term frequency – TF), the frequency of occurrence of the processing token in the indexing database (i.e., total frequency – TOTF) and the number of unique items in the database that contain the processing token (i.e., item frequency – IF, frequently labeled in other publications as document frequency – DF). With the inverse document frequency – IDF, the basic algorithm is improved by taking into consideration the frequency of occurrence of the processing token in the database.

TF-IDF is one of the most successful and well-tested techniques in Information Retrieval (IR). A document is represented as a vector of weighted terms. The computation of the weights reflects empirical observations regarding text. Terms that appear frequently in one document (TF=term-frequency), but rarely on the outside (IDF=inverse-document-frequency), are more likely to be relevant to the topic of the document. Therefore, the TF-IDF weight of a term in one document is the product of its term-frequency (TF) and the inverse of its document frequency (IDF). In addition, to prevent longer documents from having a better chance of retrieval, the weighted term vectors are normalized to unit length.

2.4.2.3. Probabilistic Vector Space Model

The items are represented with a vector of features with a probability (a real number). The probabilistic method seeks to estimate the probability that a document satisfies the information need represented by the profile. The probabilistic method is thus a generalization of the exact match technique in which we seek to rank order documents by the probability that they satisfy the information need rather than by making a shape decision. To develop this probability, term frequency information (weighted to emphasize within document frequency and to de-emphasize across-document frequency) is treated as an observation, and the distribution of the binary event "document matches profile" conditioned by that observation is computed. Bayesian inference networks have proven to be a useful technique for computing this condition probability.

Therefore, systems that classify the items with a naive Bayesian classifier (see section 2.9.4), keep the information of the items with the probabilistic vector space model method. Since it is possible to construct a Bayesian inference net that computes the cosine of the angle between two vectors, the probabilistic vector space method can be interpreted as a special case of the probabilistic method.

2.4.3. Weighted N-Grams

N-Grams can be viewed as a special technique for conflation (stemming – see section 2.7.2.1.3) and as a unique data structure in information systems. N-Grams are a fixed length consecutive series of n characters. Unlike stemming that generally tries to determine the stem of a word that represents the semantic meaning of the word, n-grams do not care about semantics. Instead they are algorithmically based upon a fixed number of characters. The searchable data structure is transformed into overlapping n-grams, which are then used to create the searchable database.

The items are represented with a net of features with weights in the nodes and edges. Based on the assumption that words tend to occur one after another a significantly high number of times, a n-gram induction method extract fixed length consecutive series of n characters and organize them with weighted links representing the co-occurrence of the different words. Therefore, the structure achieves a context representation of the words.

This model alleviated the polsemy problem (no attention is paid to word ordering or word context [Sorensen and McElligott, 1995]) as opposed to single words. Thus, not only did certain words recur in documents of interest to a user, but that they appeared in the same context as they had in items previously deemed interesting by the user.

2.4.4. Weighted Semantic Networks

A semantic networks [Potter and Trueblood, 1988] is a representational format that permit the meanings of words to be stored, so the humanlike use of these meanings is possible. In natural language names are used to identify concepts (specific or abstract) and context mechanisms to make the language frugal and concise, thus actually enhancing the expressiveness of a finite vocabulary. Although semantic data models also use logical names in order to identify externally the objects, they do not support context mechanisms. Offering contexts in semantic networks (and in data models in general) is essential, in order for their contents to be more understandable and more expressive and their management to be simpler, more flexible, and more effective by both the designers and the users.

The approach implemented by Minio and Tasso in the ifWeb system [Minio and Tasso, 1996] consists as follows: the semantic network base contains a collection of semantic networks describing typical pattern of topic of user's interest. Each semantic network includes a central node that represents a main topic and some satellite nodes connected to it through an arc that represents related topics of interests. Nodes and arcs are weighted with respect to the importance of the topic and the strength of the relationship between topics. However, the Stefani and Strapparava approach in the SiteIF system [Stefani and Strapparava, 1998] consists as follows: every node is a word or an interesting concept and the arcs between nodes are the co-occurrence relation of two words; every node and every arc has a weigh that represents a different level of interests for the user.

2.4.5. Weighted Associative Networks

An associative network consists on a set of nodes that represent the primary terms, concepts or words, in which a user is interested in. A set of weighted links establishes the organization of the terms into relevant phrases. Associative networks differ from the semantic networks because semantic networks have different generic link types such as synonymy, superclass-subclass, and also possibly disjunctive and conjunctive sets of links. Contrasting with this, associative networks (somewhat like artificial neural systems) have only a single link type, a weighted edge, the semantics being implicit in the structure of the network and the parameters associated with the processing [Riordan and Sorensen, 1995].

2.4.6. Classifiers

The systems that use a classifier as a user profile learning technique keep as a profile the structure of the classifier. This is the case of neural networks, decision trees, inducted rules and Bayesian networks.

2.4.6.1. Neural Networks

A neural network is a network of input and output cells, based upon neuron functions in the brain. It is composed of a large number of highly interconnected processing elements that are analogous to neurons and are tied together with weighted connections that are analogous to synapses. Neural networks create a compact representation that responds to queries quickly.

For instance, Jennings and Higuchi employed a neural network for constructing the users profile [Jennings and Higuchi, 1993]. During the training period, users rate documents as being interesting or not for them. For each content-bearing word that occurs at least twice in the set of training documents, a node is introduced into the neural network whose initial activity corresponds to its frequency in the positively rated documents. The link weights correspond to the co-occurrence frequency of the linked words within the same documents. When new documents are presented to the trained neural network, the nodes that correspond to the meaning-bearing words in the document become activated with their initial activity, and propagate their activity via the differently weighted links to other nodes. After a certain period of time, the overall network activity is measured and the new document rated as interesting for the user if the activity exceeds a given threshold.

2.4.6.2. Decision Trees

A decision tree is a way to classify data. It consists of a set of nodes and a set of directed edges that connect the nodes. Think of an edge as an arrow pointing from one node to another node. Consider a node N. The nodes that N points to are called its children, and N is their parent. Internal nodes are nodes that have children, and leaf nodes are nodes with no children. The internal nodes represent questions about the parameters, and the edges represent answers to those questions: values for the parameters. The leaf nodes represent a final decision.

2.4.6.3. Inducted Rules

For example, an association rule expresses the relationship that one product is often purchased along with other products. Association rules have been used for many years in merchandising, both to analyze patterns of preference across products, and to recommend products to consumers based on other products they have selected. Association rules can form a very

compact representation of preference data that may improve efficiency of storage as well as performance.

2.4.6.4. Bayesian Networks

A Bayesian network is a directed acyclic graph in which nodes represent propositional variables and arcs represent dependencies [Jensen, 1996]. A node's value is a function of the values of the nodes it depends upon. Leaf nodes represent propositions, which can usually be determined by observation. The values of nodes are determined by inference. The resulting model is very small, very fast and essentially as accurate as nearest neighbors methods [Breese et al., 1998].

A possible implementation is a Bayesian network with a node representing the information of each item in the domain. The states of each node correspond to the possible vote values for each item.

2.4.7. User-Item Ratings Matrix

Some collaborative filtering systems maintain as user profiles a user-item ratings matrix. The user-item ratings matrix contains historical ratings of the users on the items. Each cell (u,i) of the matrix contains a rating representing the evaluation of the user u to the item i, and an empty value if there is no evaluation.

These systems do not use any learning profile technique (see section 2.7.1) and they concentrate all the process in the user profile matching techniques (see section 2.10).

2.4.8. Demographic Features

Demographic filtering systems create the user profile through stereotypes. Therefore, the user profile representation is a list of demographic features that represent the kind of user. None of these systems use any learning profile technique (see section 2.7.1) and they concentrate all the process in the stereotype reasoning [Kobsa et al., 2001].

2.5. Initial Profile Generation

It is desirable to know as much as possible from the user so that the agents provide satisfactory results from the very beginning. However, the user is usually not willing to spend much time to define his interests for creating his profile. Moreover, user's interests may change over time making the profiles difficult to maintain. For these reasons, the method for the initialization and maintenance of the user profiles is a difficult aspect of the design and development of intelligent agent systems. The automation level of the acquisition of the user profiles can range from manual input, through semi-automatic procedures (stereotyping and training sets), to the automatic recognition by the agents themselves. In the latter case, automatic recognition, it is not considered an initial profile generation technique, but the initial profile starts without data (empty – see section 2.5.1) and it is constructed in a implicit way (see section 2.6.3).

Table 4 shows the initial profile generation techniques used by the different analyzed systems.

NAME	TECHNIQUE
ACR News	Training Set
Amalthaea	Manual
Anatagonomy	Empty
Beehive	Empty
Bellcore Video Recommender	Training Set
Casmir	Not Specified
Fab	Empty
GroupLens	Empty
ifWeb	Training Set, Stereotyping
InfoFinder	Training Set
INFOrmer	Training Set
Krakatoa Chronicle	Empty
LaboUr	Training Set
Let's Browse	Training Set
Letizia	Empty
LifeStyle Finder	Stereotyping
MovieLens	Training Set
News Dude	Training Set
NewsWeeder	Training Set
NewT	Training Set
Personal WebWatcher	Manual
PSUN	Training Set
Re:Agent	Manual, Training Set
Recommender	Training Set
Ringo / FireFly	Training Set
SIFT Netnews	Training Set
SiteIF	Empty
Smart Radio	Training Set
Syskill & Webert	Manual and Stereotyping
Tapestry	Empty
Webmate	Empty
WebSail	Empty
WebSell	Empty
Websift	Training Set
WebWatcher	Manual

Table 4. Initial Profile Generation Technique of the Systems

2.5.1. Empty

Some systems do not care about the initial profile, but they start with an empty profile structure (e.g., [Chen and Sycara, 1997], [Balabanovic and Shoham, 1997] and [Cunningham et al., 2001]). There is no initial phase, the profile structure is filled through an automatic recognition method when the user begins the interaction with the system.

2.5.2. Manual

The system asks users to register their interests in the form of keywords, topics and so on. One of the advantages of this method is the transparency of the system behavior. When items have been delivered to a user, the user can usually easily guess why each item was delivered. One problem with this method, though, is that it requires much effort on the part of the user. Another problem is that people cannot necessarily specify what they are interested in because their interests are sometimes unconscious.

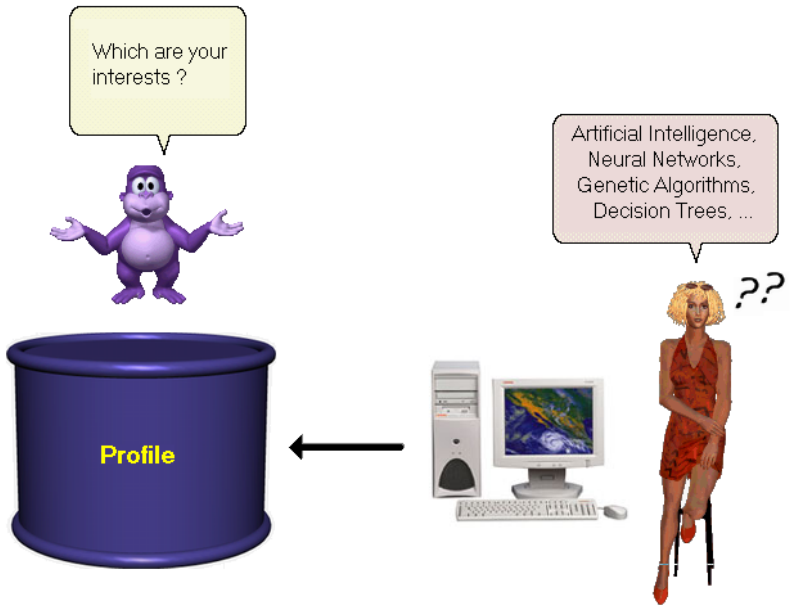

Figure 6. Manual Initial Profile Generation

Few systems use this technique, Sift Netnews [Yan and Garcia-Molina, 1995], NewT [Sheth and Maes, 1993] and Amalthaea [Moukas, 1997]. WebWatcher ([Armstrong et al., 1995] and [Joachims et al., 1997]) can be considered a special approach to this technique. Every time the user wants to use WebWatcher, the system requires that he describes his interests by means of specific words in order to adapt the profile to his needs. These words are used in every session as the initial profile.

Moreover, due to the changing interests of the user, the systems need further effort in manually update the profile. For instance, in the Sift Netnews [Yan and Garcia-Molina, 1995], when the user wants to include/exclude one of the interests contained in his profile, he has to modify it by hand. Thus, this method requires much effort on the user behalf and, therefore, the profile is less accurate. A manual technique is maybe ore suitable as an automatic method for profile definition, as it is used in Re:Agent [Boone, 1998]. In such system, the user has the option to change his profile by hand apart from the automatic procedure.

2.5.3. Stereotyping

The creation of an initial model can be regarded as a classification problem, aimed at generating initial predictions about the user [Kobsa et al., 2001]. The user model is initialized by classifying users in stereotypical descriptions [Rich, 1979], representing the features of classes of users. The use of stereotypes in computer systems for maintaining models of their users was introduced by Rich with the system Grundy [Rich, 1979]. Typically, the data used in the classification is demographic data and the user is asked to fill out a registration form: record data (name, address, phone number, etc.), geographic data (area code, city, state, country), user characteristics (age, sex, education, disposable income, etc.), psychographic data (e.g., data indicating lifestyle), user qualifying data (frequency of product/service usage, etc.) and registration for information offerings, participation in raffles, etc.

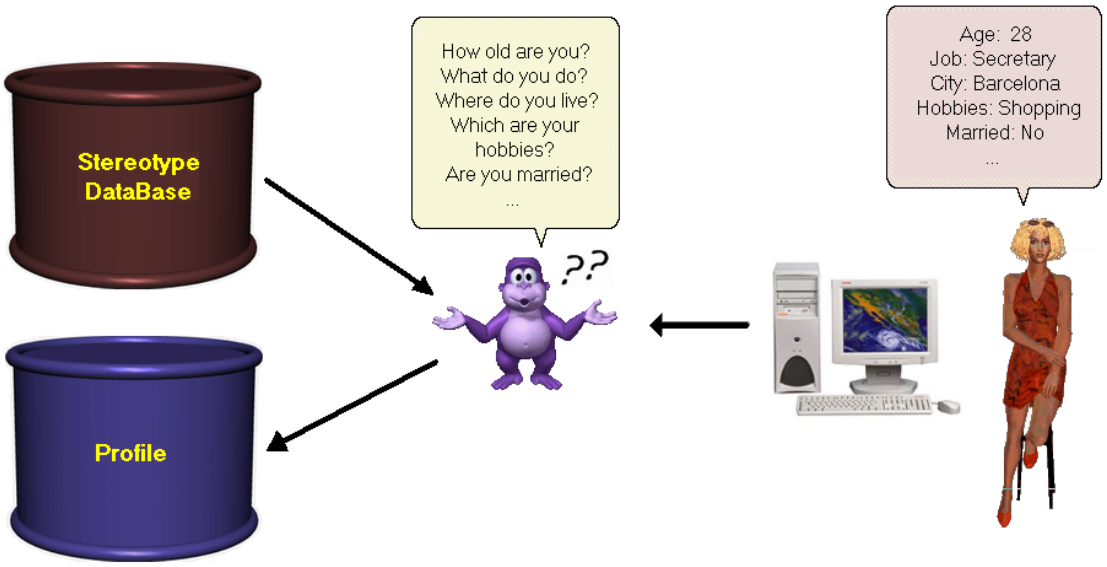

Figure 7. Initial Profile Generation through Stereotyping

An example is the method implemented by Krulwich in the LifeStyle Finder [Krulwich, 1997] which uses a commercially available database of demographic data that encompasses the interests of people nationwide. The demographic generalization approach for user profiling proposed by Krulwich consists of several steps: first, given a set of input data, the set of demographic categories to which the user is most likely to belong is computed. If only one category matches, all the data available for the category are used as a broad profile of the user, and the process ends. If more than one category matches the user data, the demographic variables whose values are similar in all the matching categories form a partial profile of the user. In this way, the demographic variable that best differentiates the matching categories can then be used to prompt the user for further information and the set of matching categories can be fed back into subsequent iterations of the algorithm to be refined. In this way, the method can converge on a single matching cluster with a close to minimal number of interactions.

Other systems which use this technique are ifWeb [Minio and Tasso, 1996], and Syskill & Webert [Pazzani et al., 1996] and [Ardissono et al., 1999].

The shortcoming of this technique is the difficulty of providing personal data by the users. Internet users normally avoid engaging in a relationship with Internet sites. This is mostly due to a lack of faith in the privacy policy of today's web sites. Normally, users either withhold personal data or provide false data.

2.5.4. Training Set

One approach is to ask the user for some explicit examples which are relevant or irrelevant for the user's interests (e.g., [Sorensen and McElligott, 1995] and [Boone, 1998]). Another approach is to ask the user for rating a set of predefined examples (e.g., [Good et al., 1999] and [Shardanand and Maes, 1994]). Once the user has given the appropriate information, the system processes the data with one of the learning techniques explained on section 2.7.

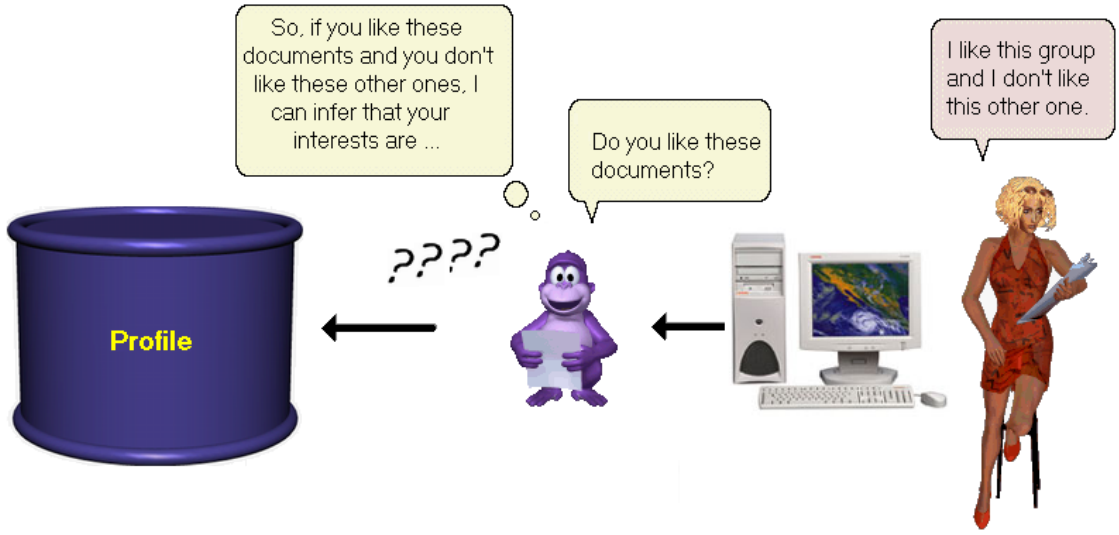

Figure 8. Initial Profile Generation through a Training Set

This mode has the advantage of simplified handling. It has the disadvantage and the danger that the selected examples are not representative and the results are less precise. Normally the learning process is of high computational complexity. Some of the systems which use this technique are ACR News [Mobasher et al., 2000], Letizia [Lieberman, 1995], FireFly [Shardanand and Maes, 1995] and LaboUr [Schwab et al., 2001].

2.6. Relevance Feedback

Human interests change as time passes. For example, a father can be very interested in baby's stuff just after childbirth, but this interest gradually decreases over time. Therefore, the user profile needs up-to-date information to update the user's interests automatically. In this section, several ways to obtain this information are presented. Then, in a next section we will see how to use this information to update the user profiles.

Typically, systems use positive information (items liked by the user) to infer the user profile. However, some systems (e.g., [Holte and Yan, 1996]) use rules for negative inference (i.e., inferring features that the user is not interested in). Authors claim that when added to their original learning apprentice, these produce a dramatic improvement in performance. Results show the new system is more than twice as effective at identifying the user's search goal and it ranks the target much more accurately at all stages of search. However, there are a few systems that cannot take into account the negative inference because the system accuracy is likely to decrease (e.g., [Schwab et al., 2000]). Thus, we can conclude that it depends on the system.

The most commonly way to obtain relevance feedback from the user are assembled in two main groups: information given explicitly for the user an information observed implicitly as from the user interaction with the Internet. Moreover, some systems propose implicit/explicit hybrid approaches. Table 5 shows the relevance of feedback techniques used by the different analyzed systems.

NAME	TECHNIQUE
ACR News	Implicit (Navigation History)
Amazon	Explicit (Ratings), Implicit (Purchase History)
Amalthaea	Explicit (Ratings)
Anatagonomy	Explicit (Ratings), Implicit (Scrolling, Enlarging)
Beehive	Implicit (Mail History)
Bellcore Video Recommender	Explicit (Ratings)
Casmir	Explicit (Ratings)
CDNow	Explicit (Ratings), Implicit (Purchase History)
Fab	Explicit (Ratings)
GroupLens	Explicit (Ratings, Text Comments), Implicit (Time Spent)
ifWeb	Explicit (Ratings)
InfoFinder	Explicit (Ratings)
INFOrmer	Explicit (Ratings)
Krakatoa Chronicle	Explicit (Ratings), Implicit (Saving, Scrolling, Time Spent, Maximizing, Resizing, Peeking)
LaboUr	Implicit (Links, Time Spent)
Let's Browse	Implicit (Links, Time Spent)
Letizia	Implicit (Links, Time Spent)
LifeStyle Finder	Explicit (Ratings), Implicit (Purchase History)
MovieLens	Explicit (Ratings)
News Dude	Explicit (Like/Dislike, I already know this, Tell me more)
NewsWeeder	Explicit (Ratings)
NewT	Explicit (Like/Dislike)
Personal WebWatcher	Implicit (Links)
PSUN	Explicit (Ratings)
Re:Agent	Nothing
Recommender	Explicit (Ratings)
Ringo / FireFly	Explicit (Ratings)
SIFT Netnews	Explicit (Like/Dislike)
SiteIF	Implicit (Links)
Smart Radio	Explicit (Ratings), Implicit (Saving)
Syskill & Webert	Explicit (Ratings)
Tapestry	Explicit (Like/Dislike, Text Comments), Implicit (Forwarding)
Webmate	Explicit (Like/Dislike)
WebSail	Explicit (Like/Dislike)
WebSell	Explicit (Not Specified)
Websift	Implicit (Navigation History)
WebWatcher	Explicit (Goal Reached), Implicit (Links)

Table 5. Relevance Feedback Technique of the Systems

2.6.1. Nothing

Some systems do not update the user profile automatically, thus, they do not need relevance feedback. All the systems that update the user profile manually (see section 2.8.2), does not need relevance feedback. Of course, neither do the systems that never modify the profile.

For instance, SIFT Netnews creates an initial profile of the user and it does not update it automatically over the time. However, the user can modify his profile by hand.

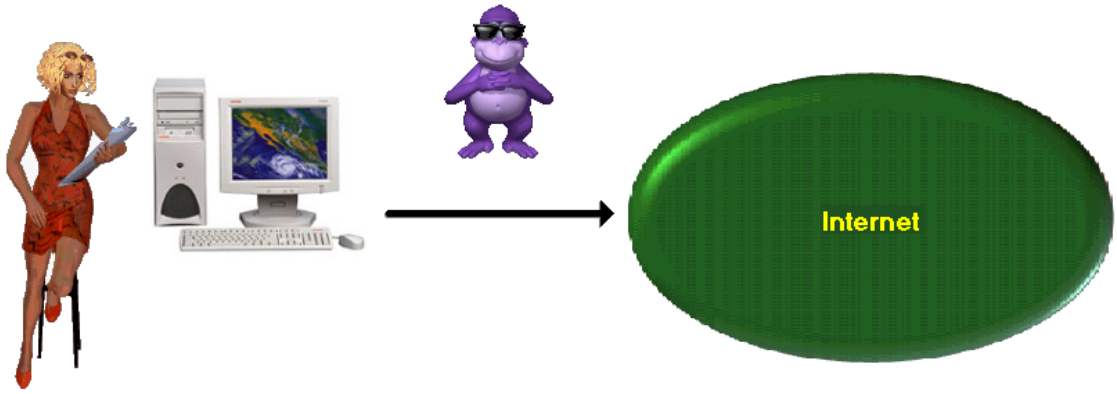

Figure 9. No Relevance Feedback

2.6.2. Explicit

In several systems, users are required to explicitly evaluate items. These evaluations indicate how relevant or interesting this item is to the user, or how relevant or interesting the user thinks a item is to other users [Rich, 1979]. Explicit feedback has the advantage of simplicity. Furthermore, in experimental systems explicit feedback has the added advantage of minimizing one potential source of experimental error, inference of the user's true reaction. Several papers exhibit the outperform of the systems achieved with the explicit relevance feedback ([Salton and Buckley, 1990] and [Buckley and Salton, 1995]).

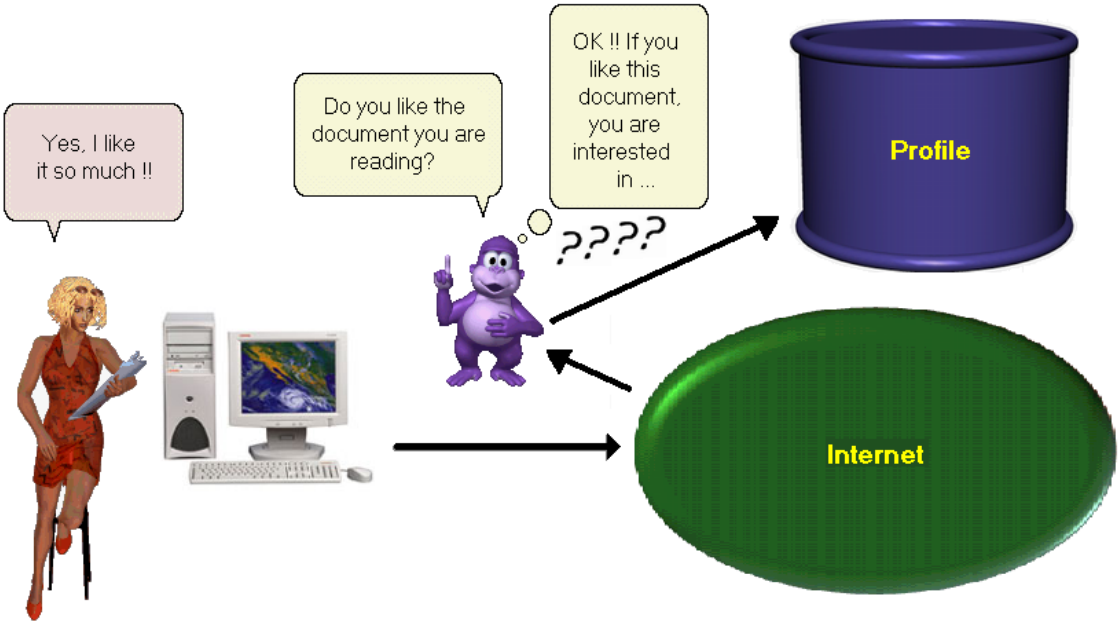

Figure 10. Explicit Relevance Feedback

But in practical applications explicit feedback has three serious drawbacks:

- First, the relevance of information is always relative to the changing information need of a user, and information environments relevance judgements of individual items are typically

assumed to be independent when in fact they are not (e.g., the third read article on the same topic may simply be rated lower because the first two items satisfied the information need and the user is judging incremental relevance at this point).

- Another problem is that numeric scales may not be well suited for describing the reactions humans have on items.

- The last problem is that computer users do not supply many ratings on the items presented to them, particularly the negative ones. Pazzani et al. report that only 15% of the users would supply interest ratings even though they were encouraged to do so [Pazzani and Billsus, 1997]. Users are generally very reluctant to perform actions that are not directed towards their immediate goals if they do not receive immediate benefits, even when they would profit in the long run [Carroll and Rosson, 1987].

We can classify the explicit relevance feedback in three groups: like/dislike, ratings and text comments.

2.6.2.1. Like/Dislike

Users are required to explicitly judge items in a binary scale, i.e., classify an object as "interesting" or "not interesting", as "relevant" or "not relevant" or as "like" or "hate". For instance, in the WebSail system [Chen et al., 2000] each document URL is preceded by two radio buttons for the user to indicate whether the document is relevant to the search query or not.

Billsus and Pazzani propose a different approach in the News Dude system [Billsus and Pazzani, 1999]. They consider that if an intelligent information agent is to be used as a personal assistant, which gradually learns about our interests and retrieves interesting information, the communications of the preferences should not limit to rate items as interesting/not interesting. For example, we might want to tell the agent that we already know about a certain topic or request information related to a certain story. Thus, the user can rate a item also with a "I already know this" or a "Tell me more".

2.6.2.2. Ratings

Classifying items with binary judgements (e.g., interesting/not interesting) sometimes is not enough, thus, systems require ratings in a discrete scale. The rating scale is typically numeric (e.g., the web bookstore Amazon.com [Amazon] offered users the opportunity to rate books in various categories on a 5-point scale) or symbolic with a mapping to a numeric scale (e.g., in Syskill&Webert [Pazzani et al., 1996] users have the possibility to rate a Web page as "hot", "lukewarm", or "cold").

2.6.2.3. Text Comments

Several sites encourage text comments from their users (e.g., Grouplens [Resnick et al., 1994] and Tapestry [Goldberg et al., 1992]). Systems gather comments about a single item and present these as a means to facilitate the decision-making process. While text comments are helpful, they require a fair amount of processing by the targeted user. The user must read each paragraph and interpret to what degree it is positive or negative.

2.6.3. Implicit

Implicit feedback means that the system automatically infers passively the user's preferences from monitoring the user's actions. [Chatterjee et al, 1998] prove empirically that the user interests can be inferred from his behavior. This is mainly due to the fact that motivating web consumers to provide personal data is proving very difficult. Users are unlikely to engage in additional efforts even when they know that they would profit in the long run [Carrol and Rosson, 1987]. Conclusions about user's interest should therefore not rely very much on user explicit feedback, but rather take passive observations about users into account as far as possible.

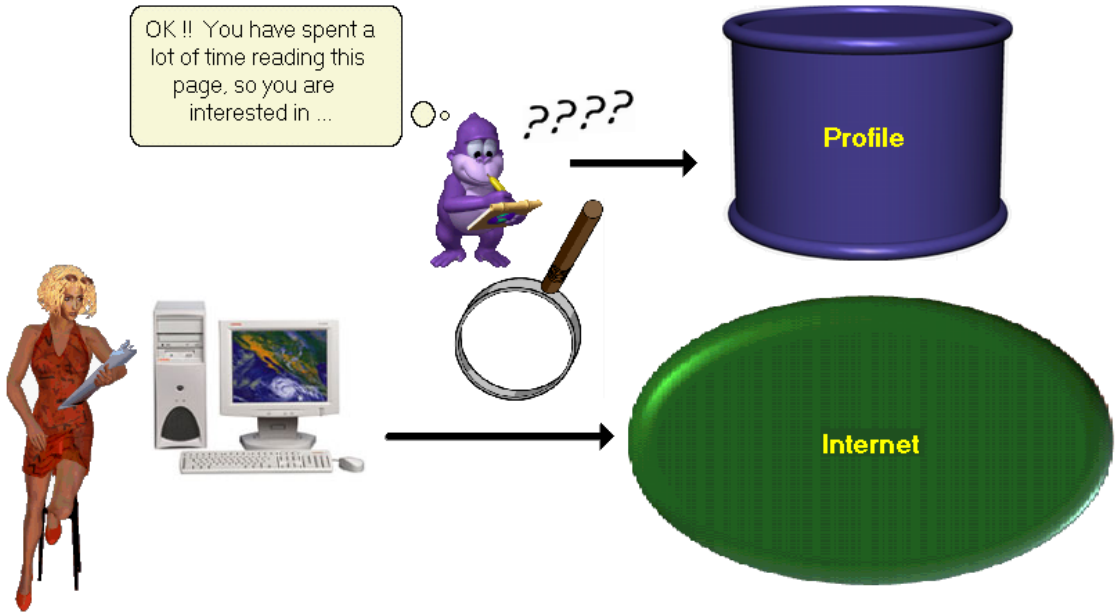

Figure 11. Implicit Relevance Feedback

The implicit feedback was early defined by [Rich, 1979], and the first system was implemented by [Mitchell et al., 1985]. Since then, a lot of systems implement implicit user profile learning in their approaches (e.g., [Stefani and Strappavara, 1998] and [Schwab et al., 2001]) and, even, some systems combined it with the explicit feedback ones (hybrids, see section 2.6.4).

The implicit methods mostly used in the state of the art to obtain relevance feedback from the user are analyzing the followed links, a history of purchases, web navigation or e-mails and the time spent in a particular web page.

2.6.3.1. Links

In the World Wide Web environment, when a user click on a link makes a choice, if competitive links are available on the current page. Hyperlinks whose documents were visited by the user are considered to be positive examples and all the others negative ones of its interests (e.g., [Lieberman, 1995] [Mladenic, 1996]). The idea is that all hyperlinks were presented to the user and the user chose to visit some of them that met his interests. For example, on an e-commerce site the user may select one of the products offered on a page to read a more detailed description. Such selective actions can be regarded as indicators of interests and preferences. For instance, WebWatcher [Joachims et al., 1997] monitors link selection on Web pages to annotate the most relevant links on each page.

Another aspect to take into account is that if the user returns immediately without having either saved the target document, or followed further links, an indication of disinterest can be assumed. Thus, the time wasted exploring the documents (see section 2.6.3.3) combined with the selected links can improve dramatically the results [Lieberman, 1995].

Some papers claim that in a general approach the assumption that links not selected are negative examples is not valid (e.g., [Schwab et al., 2000] and [Schwab et al., 2001]). It is a common situation that objects are overlooked, and it is impossible to have an overview of all relevant objects. Sometimes pages that are not visited at the moment may be visited at a later point, and sometimes they are ignored forever even when the user is interested in them since it is too time consuming or simply not possible to follow every interesting link. Therefore, classifying the objects not visited as negative examples seems to be a dangerous assumption.

2.6.3.2. History

Purchase history in e-commerce (e.g., [Amazon], [CDNow] and [Krulwich, 1997]), navigation history in WWW (e.g., [Cooley et al., 1999] and [Mobasher et al., 2000]) or mail boxes in e-mail (e.g., [Huberman and Kaminsky, 1996]) are generally regarded as strong indicators for user's interests. Analyzing the content of the items contained in the history, we can get relevant information representing the user's interests.

For instance, in e-commerce, if the customer relationship application uses an underlying feature-based model, the assumption is made that a purchase is a strong indicator of interest in some of the features of the purchased product. Of course, there is no one-to-one mapping of purchases and interests since, for example, customers purchase items for other people (e.g., as gifts) and because people may already own an available item. Amazon attempts to address this issue by disregarding purchases with shipping addresses that are different from the user's address, and by encouraging customers to indicate that they already own a particular item.

2.6.3.3. Time Spent

[Morita and Shinoda, 1994] applied statistical analysis on the collected data and concluded that a major factor that influence the time spent for an article is the preference of the user for the article. The results from their analysis concluded that there is a strong tendency to spend a long time to read articles that are rated interesting and to spend little time on not interesting articles. They discovered that interpreting as "interesting" articles, on which the reader spent more than 20 seconds reading produced better recall and precision (see sections 2.11.2.2 and 2.11.2.3) in a text filtering experiment than using documents explicitly rated by the user as interesting.

[Konstan et al., 1997] initial studies show that we can obtain substantially more ratings by using implicit ratings. Their results point and that predictions based on time spend reading are nearly as accurate as predictions based on explicit numerical ratings. They also provide large-scale confirmation of the work of Morita and Shinoda in finding the relationship between time and rating without regard for the length of the article holds true.

Sakagami and Kamba claim that the time spent is intuitively reasonable because we tend to spend more time reading interesting articles than uninteresting ones [Sakagami and Kamba, 1997]. In their experiment, however, they asked the subjects "to do nothing but read articles", that is "not do other things such as leaving the terminal a while to get a cup of coffee or reading newly arrived e-mail messages". We cannot generalize these experimental results to real-world settings where users are distracted and interrupted [Oard and Marchionini, 1996], since measurement of effective viewing time is difficult. It is often impossible to tell whether the user

has been present in front of the computer screen and looked at a specific item within a specific time interval. These conditions show the limitations of their method. In actual situations, we often receive e-mail and telephone calls and we are subjected to other interruptions. Therefore, the time spent method is not sufficiently practical.

However, viewing time can serve as negative evidence [Kobsa et al., 2001]. If the presentation time (and thus the maximal viewing time) of a document is below a certain threshold, then the information on that page is most likely to be not interesting to the user. For instance, if the download of a Web is aborted or if the user presses the Back button shortly after the page download commenced, this may be regarded as an even stronger indicator that the user is actually not interested in the item just selected (provided that the download time was within an acceptable limit).

2.6.3.4. Others

There are many other examples for confirmatory actions. For documents like Web pages, news articles or e-mail messages, it is interesting to monitor whether the user does any further processing action. For example, saving a document ([Kamba et al., 1995]), printing a document, bookmarking a Web page, deleting a document, replying or forwarding an e-mail [Goldberg et al., 1992], or scrolling, maximizing, minimizing or resizing the window containing the document or the Web page ([Kamba et al., 1995], [Sakagami et al., 1997]). Since these actions are performed under the control of the application, they can be registered and evaluated to learn the user profile.

However, [Kobsa et al., 2001] do not recommend a universal logging of usage data on the micro-interaction level, such as the tracking of mouse movements within applets, unless the purpose of the login has already been specified (e.g., for determining user's interest in page segments, like in systems of [Sakagami et al., 1997]). The amount of data collected is very large, the computation needed to derive recommendations for adaptations is extensive, and the confidence in the suitability of these adaptations is likely to be relatively low. However, it seems promising to experiment with such data in smaller, laboratory contexts to drive the development of new methods in this area.

2.6.4. Hybrid

The limited evidence available suggests that implicit feedback has great potential but its effectiveness remains unproven. As it is common in many technologies the best performing system results of combining several existing technologies, in this field implicit feedback can be combined with existing explicit feedback systems to form a hybrid system (see Figure 12). Providing implicit feedback greatly decreases the user's efforts, whereas providing explicit feedback helps the system to infer user preferences accurately.

One approach of such combination is to use implicit data as a check on explicit ratings [Nichols, 1997]. For instance, if an evaluator is explicitly rating an item then there should be some implicit data to confirm that he has actually examined it. If there is no evidence to suggest that the evaluator has examined an item then perhaps their rating should be ignored, or reduced in importance. Conversely, an evaluation with a relatively long "examine time" may be increased in importance.

A different case is Anatagonomy [Sakagami et al., 1997]. Giving explicit feedback is optional, and it should only be used when they wish to show explicit interest. WebWatcher [Joachims et al, 1997], LifeStyle Finder [Krulwich, 1997], Krakatoa Chronicle [Kamba et al., 1995],

GroupLens [Resnick et al., 1994], CDNow [CDNow] and Amazon [Amazon] also use hybrid relevance feedback.

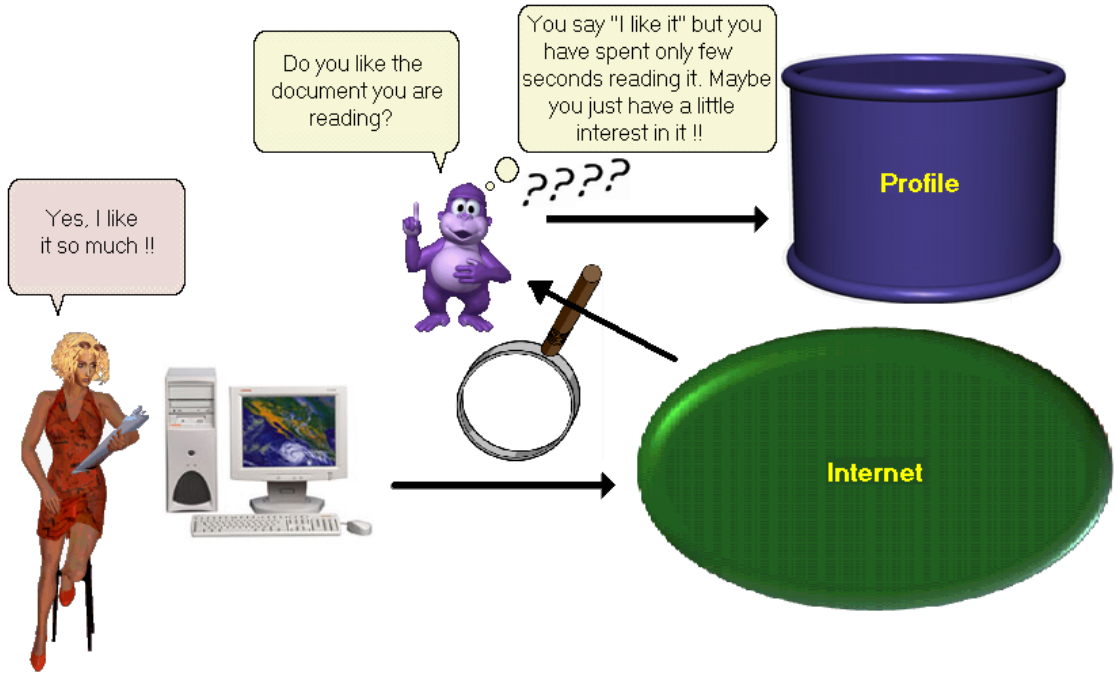

Figure 12. Hybrid (Implicit/Explicit) Relevance Feedback

2.7. Profile Learning Techniques

The previous section describes sources of information that are potentially representative of the user interests. This section details several techniques to build a user profile through these data. These techniques can be seen as a previous step to represent the user profile. Typically, the relevance feedback is processed to obtain the general preferences of the user. However, not all the systems apply a learning method to build a profile. Some systems just keep as a profile the relevance feedback without any processing.

Besides, when the relevance feedback is composed by text without structure, it is necessary a first step before learning a profile. It consists in to apply some information retrieval technique to extract structured relevant information. Some systems just use an information retrieval technique to learn a profile and represent it as a structure of indexed words, although the information retrieval techniques cannot be considered artificial intelligence techniques, since they just index words.

Some systems have an off-line phase during which they learn a model of a user behavior, and then an online phase during which they apply the model in real time. Most systems, however, use a lazy learning approach (online), in that they build and update the model while making recommendations in real time. Offline learning methods may prove practical for environments in which knowledge of consumer preferences changes slowly with respect to the time needed to build the model but are not suitable for environments in which consumer preference models must be updated rapidly or frequently.

This section is structured as follows. First, the typical systems that need no profile learning techniques are briefly explained. Then, since the relevance feedback of some systems is composed by text, the information retrieval techniques used in these systems are summarized. Finally, the most commonly profile learning techniques are reviewed: data mining and classifiers. Table 6 shows the profile learning techniques used by the different analyzed systems.

NAME	TECNIQUE
ACR News	Data Mining (Induction Rule Learning, Clustering)
Amazon	Not Necessary
Amalthaea	Feature Selection (Stemming), Frequency Vector Space Model Method (TF-IDF)
Anatagonomy	Frequency Vector Space Model Method (TF-IDF)
Beehive	Data Mining (Clustering)
Bellcore Video Recommender	Not Necessary
Casmir	Simple Positive Reinforcement, Simple Positive Reinforcement with Query Keyword Overriding, Positive and Negative Reinforcement, Positive and Negative Reinforcement with Query Keyword Overriding
CDNow	Not Necessary
Fab	Frequency Vector Space Model Method (TF-IDF)
GroupLens	Not Necessary
ifWeb	Feature Selection (Stop-Words, Stemming, …)
InfoFinder	Feature Selection (Heuristics), Decision Tree (ID3)
INFOrmer	Feature Selection (Stop-Words, Stemming, …)
Krakatoa Chronicle	Frequency Vector Space Model Method (TF-IDF)
LaboUr	Feature Selection (Pruning, Weighting Words), Boolean Vector Space Model Method
Let's Browse	Frequency Vector Space Model Method (TF-IDF)
Letizia	Frequency Vector Space Model Method (TF-IDF)
LifeStyle Finder	Not Necessary
MovieLens	Frequency Vector Space Model Method (TF-IDF), Data Mining (Induction Rule Learning - Ripper)
News Dude	Short Term: Frequency Vector Space Model Method (TF-IDF), Long Term: Boolean Vector Space Model Method
NewsWeeder	Frequency Vector Space Model Method (TF-IDF), MDL
NewT	Feature Selection (Stop-Words, Stemming), Frequency Vector Space Model Method (TF-IDF)
Personal WebWatcher	Frequency Vector Space Model Method (TF-IDF)
PSUN	Feature Selection (Stemming), N-Gram Induction (Schank, Hebian Learning and Minds & Minsky)
Re:Agent	Feature Selection (Stop-Words), Frequency Vector Space Model Method (TF-IDF), Data Mining (Clustering), Neural Network
Recommender	Data Mining (Induction Rule Learning - Ripper)
Ringo / FireFly	Not Necessary
SIFT Netnews	Boolean Vector Space Model Method, Frequency Vector Space Model Method (TF-IDF)
SiteIF	Feature Selection (Stop-Words, Stemming, …)
Smart Radio	Not Necessary
Syskill & Webert	Feature Selection (Stop-Words), Boolean Vector Space Model Method, Frequency Vector Space Model Method (TF-IDF), Decision Tree (ID3)
Tapestry	Not Necessary
Webmate	Frequency Vector Space Model Method (TF-IDF)
WebSail	Frequency Vector Space Model Method (TF-IDF)
WebSell	Not Necessary
Websift	Data Mining (Inducted Rule Learning)
WebWatcher	Frequency Vector Space Model Method (TF-IDF), Winnow, WordStat, Random

Table 6. Profile Learning Technique of the Systems

2.7.1. Not Necessary

Some systems keep as a user profile the information directly acquired from the system, thus, they do not need a profile learning technique. Mainly, three kinds of systems do not need a profile learning method:

- Systems that acquire the information of the user profile from a database. For instance, electronic commerce systems ([Amazon], [CDNow], [Cunningham et al., 2001]) that extract the information from a database of products and keep as a profile a purchase list (see section 2.4.1).

- Collaborative filtering systems ([Goldberg et al., 1992], [Resnick et al., 1995], [Shardanand and Maes, 1995]) that keep as a profile a matrix with the user-item ratings (see section 2.4.7).

- Systems that create an initial profile through stereotyping (see section 2.5.3) and do not modify it ([Krulwich, 1997]). This is the case of demographic filtering systems (see section 2.3.1).

The systems that do not need a profile learning technique concentrate the information filtering tasks on the profile-item or profile-profile matching techniques.

2.7.2. Information Retrieval Techniques

Typically, the source of the information to generate a user profile is not structured, but it is a text document like an e-mail, an electronic new or a Web page. A technique to extract relevant information from the unstructured text documents is needed. Thus, the information retrieval techniques are suitable since they automate the process of examining text documents to extract structured relevant information. Such process is based in two main steps: feature selection and information indexing.

2.7.2.1. Feature Selection

A problem with observation data is that the dimensionality of the structures describing the document still is rather large. Learning under these conditions is not practical, because the amount of data needed to approximate a concept in d dimensions grows exponentially with d. Hence there is a need for dimensionality reduction. If we decide to ignore all the additional information and use the statistical indexing approach (see section 2.7.2.2), we still end up with several tens of thousands of different words that occur in our documents. Not only is using all these words time-consuming but also many of them are not really important for our learning task.

Furthermore, Schwab et al. claim that every user has different interests and, therefore, also different features are important to her. In this way, feature selection should be individualized and be performed individually for each user [Schwab et al., 2001].

There are several approaches to reduce number of different words: stop-words, pruning, stemming, word weighting and latent semantic indexing.

2.7.2.1.1. STOP-WORDS

In the text documents normally there are a list of frequently occurring words that typically are not very relevant to classification problems [Kowalski, 1997]. Words on the stop list (e.g., the, is, very, and if) are always excluded from consideration as informative words ([Riordan and Sorensen, 1995], [Stefani and Strappavara, 1998], [Pazzani et al., 1996]).

2.7.2.1.2. PRUNING

Pruning words can be considered as an evolution of the Stop-Words approach. In this case, apart from exclude frequent and not relevant words from the text, the infrequent words are also excluded ([Cohen, 1995a], [Asnicar and Tasso, 1997], [Schwab et al., 2001]).

2.7.2.1.3. STEMMING

Conflation is the term frequently used to refer to mapping multiple morphological variants to a single representation (stem). The premise is that the stem carries the meaning of the concept associated with the word and the affixes (endings) introduce subtle modifications to the concept or are used for syntactical purposes. Languages have precise grammars that define their usage, but also evolve based upon human usage. Thus exceptions and non-consistent variants are always present in languages that typically require exception look-up tables in addition to the normal reduction rules. Stemming algorithms are used to improve the efficiency of the information system and to improve recall (see section 2.11.2.2). Several systems use this approach ([Balabanovic and Shoham, 1995], [Moukas, 1997], [Riordan and Sorensen, 1995], [Asnicar and Tasso, 1997], [Sorensen and McElligott, 1995] and [Stefani and Strapparava, 1998]).

2.7.2.1.4. WORD WEIGHTING

Many approaches introduce some sort of word weighting and select only the best words ([Armstrong et al., 1995][Pazzani et al., 1996][Mladenic, 1996]). For instance, Schwab et al. weight the words with the probability of relevance of the feature. The weights are recalculated with the time to adapt to the changing interests of the user. Therefore, the feature selection changes with the time too [Schwab et al., 2001].

2.7.2.1.5. LATENT SEMANTIC INDEXING (LSI)

Latent Semantic Indexing [Deerwester et al., 1990][Foltz, 1990] is based on the assumption that there is an underlying or "latent" structure represented by interrelationships between words [Kowalski, 1997].
The idea is to represent the documents with a description on a more abstract level. LSI takes advantage of the implicit higher-order structure of the association of terms with articles to create a multi-dimensional semantic structure of the information. Through the pattern of co-occurrences of words, LSI is able to infer the structure of relationships between articles and words. Singular-value decomposition (SVD) of the term by article association matrix is computed producing a reduced dimensionality matrix containing the best K orthogonal factors to approximate the original matrix as the model of "semantic" space for the collection. This semantic space reflects the major associative patterns in the data while ignoring some of smaller variations that may be dues to idiosyncrasies in the word usage of individual documents. In this way, LSI produces a representation of the underlying "latent" semantic structure of the information.

2.7.2.1.6. OTHERS

Krulwich and Burkey Heuristics used heuristics to extract significant phrases from the document text [Krulwich and Burkey, 1995]. These heuristics are based on the observation that document authors tend to use syntactic methods to delineate key phrases or ideas in documents, such as putting them in italics, identifying them with acronyms, or the like. Some of the Machine Learning techniques for feature selection could also be used [Caruana and Freitag, 1994], but most of them take too long in situations with several tens of thousands of features.

2.7.2.2. Information Indexing

The items are typically represented by some structure of features. The features are terms, usually words or concepts that appear in the documents. Associated with each feature there is a value (Boolean or real) representing its presence or relevance. Three main information indexing paradigms can be identified in the Information Retrieval literature: Statistical Indexing, Semantic Indexing and Contextual/Structural Indexing. Statistical Indexing uses frequency of occurrence of words to calculate the potential relevance of an item. Semantic Indexing characterizes the documents and queries so as to represent the underlying meaning. It emphasizes natural language processing or the use of AI-like frames. Contextual/Structural Indexing takes advantage of the structural and contextual information typically available in retrieval systems.

Based on these three paradigms, information structures are extracted. The most commonly structure used to represent the items is vectors (see section 2.4.2), but several approaches have proved the useful of a more complicated structure like a network (see sections 2.4.3, 2.4.4 and 2.4.5).

2.7.3. Data Mining

As merchandisers gained the ability to record transaction data, they started collecting and analyzing data about consumer behavior. The idea is to identify potentially useful information implicit in these records. The term data mining is used to describe the collection of analysis techniques used to infer rules from or build models from large data sets.

These techniques have been used during years with important benefits to the databases of the traditional commercial enterprises. Two main goals of these techniques are to save money by discovering the potential for efficiencies, or to make more money by discovering ways to sell more products to customers. For instance, companies are using data mining to discover which products sell well at which times of year, so they can manage their retail store inventory more efficiently. Other companies are using data mining techniques to discover which customers will be most interested in a special offer, reducing the costs of direct mail or outbound telephone campaigns. The idea is to apply these techniques to the electronic commerce with the same purposes.

Typically, data mining has two phases: the learning phase and the use phase. The learning phase, the data mining system analyzes the data and builds a model of consumer behavior (e.g. association rules). This phase is often very time-consuming and may require the assistance of human analysts. Thus, these techniques may prove practical for environments in which knowledge of consumer preferences changes slowly with respect to the time needed to build the model but are not suitable for environments in which consumer preference models must be updated rapidly or frequently. After the model is build, the system enters a use phase where the model can be rapidly and easily applied to consumer situations.

When we want to apply the data mining techniques in the profiling field, some questions come up in our minds: Is there useful data (i.e., preferences) hidden in the activity records? Can the data be extracted accurately and efficiently? Is the extracted data of high quality? Several papers prove that the data mining techniques, well-known in other fields, are also very useful in this field ([Etzioni, 1996], [Cooley et al., 1999] and [Mobasher et al., 2000]).

There are a lot of data mining techniques, but the most commonly used are induction rule learning and clustering.

2.7.3.1. Induction Rule Learning

One of the best-known examples of data mining techniques is the discovery of association rules by inductive learning. The association rule discovery methods initially find groups of items occurring frequently together in many transactions. Such groups of items are referred to as frequent item sets [Mobasher et al., 2000]. Association rules capture the relationships among these items based on their patterns of co-occurrence across transactions.

The number of possible association rules grows exponentially with the number of items in a rule, but constraints on confidence and support, combined with algorithms that build association rules with itemsets, reduce the effective search space. They are more commonly used for larger populations rather than for individual users.

Some examples of inductive learning techniques are Ripper [Cohen, 1995b], Slipper [Cohen and Singer, 1999], CN2 [Clark and Niblett, 1989] and C4.5rules [Quinlan, 1994].

2.7.3.2. Clustering

Traditional collaborative filtering techniques are often based on matching the current user profile against clusters of similar profiles obtained by the system over time from other users (see section 2.10.3). A similar technique can be used in the context of Web personalization by first clustering user transactions. However, in contrast to collaborative filtering, clustering user transactions based on mined information from access logs does not require explicit ratings or interaction with users. Standard clustering algorithms generally partition the transactions space into groups of items that are close to each other based on a measure of distance. In section 2.10.3 there is a brief survey of clustering techniques applied to user clustering into collaborative filtering.

2.7.4. Classifiers

Classifiers are general computational models for assigning a category to an input. Classifiers have been quite successful in a variety of domains ranging from the identification of fraud and credit risks in financial transactions to medical diagnosis to intrusion detection. To build a recommender system using a classifier is to use information about the item and the user profile as the input, and to have the output category represent how strongly to recommend the item to the user. Classifiers may be implemented using many different machine learning strategies include neural networks (see section 2.4.6.1), decision trees (see section 2.4.6.2) and Bayesian networks (see section 2.4.6.4).

2.7.4.1. Neural Networks Learning

Learning in neural networks is achieved by training the network with a set of data. Each input pattern is propagated forward through the network and active output cells represent the interest of the user. When an error is detected it is propagated backward adjusting the cell parameters to reduce the error, thus achieving learning. Neural networks can be considered as function approximators based on sums of nonlinear, typically sigmoidal, basis functions. This technique is very flexible and can accommodate a wide range of distributions. A major risk of neural networks is that they can overfit by learning the characteristics of the training data set and not be generalized enough for the normal input of items. In applying training to a neural network approach, a validation set of items is used in addition to the training items to ensure that overfitting has not occurred. As each iteration of parameter adjustment occurs on the training set, the validation set is retested. However, because backpropagation is a gradient descent algorithm, they can be slow to train.

2.7.4.2. Decision Trees Learning

Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. The learned trees can also be represented as a set of if-then rules. Decision tree learners build a decision tree by recursively partitioning examples into subgroups until those subgroups contain examples of a single class. A partition is formed by a test on some attribute (e.g., is the feature database equal to 0). The learner selects the test that provides the highest gain in information content. The most used decision tree learner applied to the profiling is the ID3 [Quinlan, 1983].

2.7.4.3. Bayesian Networks Learning

A Bayesian network learner algorithm is applied to a set of training data, searching over various model structures in terms of dependencies for each item. In the resulting network, each item will have a set of parent items that are the best predictors of its votes. A decision tree encoding the conditional probabilities for that node represents each conditional probability table. The model can be build off-line over a matter of hours or days. Thus, this technique may prove practical for environments in which knowledge of consumer preferences changes slowly with respect to the time needed to build the model but are not suitable for environments in which consumer preference models must be updated rapidly or frequently.

2.7.5. Inductive Logic Programming (ILP)

Inductive logic programming lies at the intersection of machine learning and computational logic, as used in logic programming. It combines inductive machine learning with the representations of computational logic. Computational logic is a more powerful representation language than the classical attribute-value representation typically used in machine learning. This representational power is useful in the context of learning user preference models, because in this way more complex types of user preferences can be detected and described. Another advantage of inductive logic programming is that it enables the use of background knowledge in the induction process. An ILP system takes as input examples and background knowledge and produces hypotheses as output. There are two forms of induction: Predictive induction starts from a set of classified examples and a background theory, and the aim is to induce a theory that will classify all the examples in the appropriate class. Descriptive induction starts from a set of unclassified examples, and aims at finding a set of regularities that hold for the examples. The advantages and disadvantages of ILP in user preference modeling are discussed in [Dastani et al., 2000].

2.7.6. Others

Several systems implemented different techniques and exhibit their performance in personalized environments. This is the case of Lang, which applied a Minimal Description Length in his NewsWedeer recommendation system. This technique is a tradeoff between model complexity and training error [Lang, 1995]. Pazzani et al. learn the user profile of Syskill&Webert with the TF-IDF approach, but they also used Winnow, WordStat and Random approach to compare the results [Pazzani et al., 1996]. Winnow learns a Boolean concept represented as a single linear threshold function of the instance features. Weights for this threshold function are learned using a multiplicative update rule. WordStat attempts to make a prediction whether a link is followed based directly on the statistics of individual words. Finally, they also introduced a random approach. It consists in a random choice of one link on the page with uniform probability. Its is typically used to provide a baseline measure against which to compare other techniques.

2.8. Profile Adaptation Techniques

Since personalized systems typically involve interaction over long periods of time, user interests cannot be assumed to stay constant.

NAME	TECHNIQUE
ACR News	Add New Information
Amalthaea	Natural Selection, Gradual Forgetting Function
Anatagonomy	Add New Information
Beehive	Add New Information
Bellcore Video Recommender	Add New Information
Casmir	Add New Information
Fab	Natural Selection
GroupLens	Add New Information
ifWeb	Gradual Forgetting Function
InfoFinder	Add New Information
INFOrmer	Add New Information
Krakatoa Chronicle	Add New Information
LaboUr	Gradual Forgetting Function
Let's Browse	Add New Information
Letizia	Add New Information
LifeStyle Finder	Add New Information
MovieLens	Add New Information
News Dude	Short-term and Long-term Models
NewsWeeder	Add New Information
NewT	Natural Selection
Personal WebWatcher	Add New Information
PSUN	Natural Selection
Re:Agent	Manual
Recommender	Add New Information
Ringo / FireFly	Add New Information
SIFT Netnews	Manual
SiteIF	Gradual Forgetting Function
Smart Radio	Add New Information
Syskill & Webert	Add New Information
Tapestry	Add New Information
Webmate	Add New Information
WebSail	Add New Information
WebSell	Add New Information
Websift	Add New Information
WebWatcher	Add New Information

Table 7. Profile Adaptation Technique of the Systems

This normally means that the most recent observations represent the current user's interests better than older ones. Therefore, there is a need of a technique to adapt the user profile to the new interests and to forget the old ones. This is essential if more and more people are to use it.

There are several approaches to adapt the user profile to the new interests: manually, just adding the new information, with a time window, aging examples, combining a short-term and a long-term model, a gradual forgetting function or the natural selection for ecosystems of agents. Table 7 shows the profile adaptation techniques used by the different analyzed systems.

2.8.1. Nothing

Some systems do not care about the profile adaptation (specially the first ones), they assign an initial profile to the user and keep it unchanged over time. Particularly this was the case of systems that assigned an initial profile using the stereotyping technique and this was not updated any longer. State of the art implementations do not use this approach, for this reason no examples can be referenced.

2.8.2. Manual

In some systems, the user has to change the profile when he is interested in updating it. For instance, in the Sift Netnews [Yan and Garcia-Molina, 1995], when the user wants to include/exclude one of the interest contained in his profile, he has to modify it by hand. Thus, this method requires much effort on the part of the user and, therefore, the profile is less accurate.

Like the manual initial profile generation (see section 2.5.2), this approach has two important problems: it requires much effort on the part of the user and people cannot necessarily specify what they are interested in because their interests are sometimes unconscious. Therefore, the manual updating turns out to be difficult when the requirements change quickly.

2.8.3. Add New Information

This approach is the most commonly used in the current systems, however it does not forget the old interests. The idea is to update the user profile adding to it the new information extracted from the user relevance feedback (see section 2.6). Thus, the profile is adapted to new user's interests, but the old ones are not forgotten.

2.8.4. Time Window

It's the most frequently used approach to deal with the problem of forgetting old interests. It consists in learning the description of user's interests only from the latest observations. The training examples are selected from a so-called time window, i.e. only the last examples are used for training [Mitchell et al., 1994]. An improvement of this approach is the use of heuristics to adjust the size of the window according to the current predictive accuracy of the learning algorithm [Widmer and Kubat, 1996].

2.8.5. Aging Examples

Maloof and Michalski implemented a variation of the time window approach [Maloof and Michalski, 2000]. Instances that are older than a certain age are deleted from the partial memory. Like the time window, the system only take into account the last examples, however, this approach totally forgets the observations that are outside the given window or older than a certain age.

2.8.6. Short-Term and Long-Term Model

Billsus and Pazzani developed an original approach to handle the profile adaptation [Billsus and Pazzani, 1999]. The originality of this approach is the use of a dual user model consisting of both a short-term and a long-term model of the user's interests. The method employs the short-term model first, because it is based on the most recent observations only. Then, the system allows the user to track news threads that have previously been rated and can label stories as already known. If a story cannot be classified with the short-term model, the long-term model is used. If the long-term model decides that the story does not contain sufficient evidence to be classified, a default score is assigned. This hybrid user model is useful in domains where the long-term user's interests are quite broad and short-term interests change fast, as is the case for news stories. Anyway, the short-term model can be considered a time window system (see section 2.8.4) with the newest observations, and the long-term model as a classic user model without adaptation to the new interests.

2.8.7. Gradual Forgetting Function

The concept was introduced by [Webb and Kuzmycz, 1996] and the main idea behind it is that the natural forgetting is a gradual process.. Therefore, a gradual forgetting function can be defined. It should produce a weight for each observation according its location in the course of time. They suggest a data aging mechanism that places an initial weight of 1 on each observation. A set proportion discounts the weight of every observation each time another relevant observation is incorporated into the model. Thus, the most recent observations become more "important" for the learning algorithms, assuming that they better represent the current users' interests than the older ones. Hence, the system becomes more noise resistant without losing its sensitivity to real changes in interest [Schwab et al., 2001]. Koychev proposes a linear gradual forgetting function [Koychev, 2000], but it can be approximated with a any function (e.g., logarithmic or exponential).

2.8.8. Natural Selection

The natural selection approach is associated with the systems that implement an ecosystem architecture of agents based on genetic algorithms (see section 2.12.2). An ecosystem of specialized agents competes in parallel giving recommendations to the user. The ecosystem evolves in the following way: the agents that produce best results are reproduced with the crossover and mutation operators and the other ones are destroyed.

2.9. User profile – Item Matching

Once the user profile containing the user preferences is created, the next step is to exploit it. Typically, the user profile is used to recommend new items considered relevant to the user. Content-based filtering systems use a direct comparison between the user profile and the new items. Thus, a user profile – item matching technique is needed. Several techniques are studied with the objective to automate the process of classifying items through its content in relevant/not relevant by computing comparisons between the representation of the user's interests and the representation of the items. This automated process is successful when it produces results similar to those produced by human comparison of the documents themselves with the actual information need.

Typically, the user profile – item matching techniques used are: a simple keyword matching, the cosinus similarity, the CBR, the Naive Bayesian Classifier, the nearest neighbor and typical classifiers. Table 8 shows the user profile – item matching techniques used by the different analyzed systems.

NAME	TECHNIQUE
ACR News	Itemset and Cluster Similarity Matching
Amalthaea	Cosinus Similarity
Anatagonomy	Cosinus Similarity
Casmir	Pre-Search Request Based Collaboration, Pot-Search Informing
Fab	Cosinus Similarity
ifWeb	Standard Keyword Matching
InfoFinder	Boolean Search Query String
INFOrmer	Graph Comparison
Krakatoa Chronicle	Cosinus Similarity
LaboUr	Bayessian Classifier & Nearest Neighbor
Let's Browse	Cosinus Similarity
Letizia	Cosinus Similarity
MovieLens	Cosinus Similarity, Inducted Rules
News Dude	Short-Term: Nearest Neighbor (Cosinus Similarity), Long-Term: Naive Bayesian Classifier
NewsWeeder	Cosinus Similarity
NewT	Cosinus Similarity
Personal WebWatcher	Naive Bayesian Classifier
PSUN	Graph Comparison
Re:Agent	Nearest Neigbour, Neural Network
Recommender	Inducted Rules
SIFT Netnews	Dot Product
SiteIF	Standard Keyword Matching
Syskill & Webert	Naive Bayesian Classifier, Nearest Neigbor, PEBLS, Cosinus Similarity, Decision Tree
Webmate	Cosinus Similarity
WebSail	TW2
WebSell	CBR with Nearest Neighbor (Pearson r Correlation)
Websift	Inducted Rules and Pattern Matching
WebWatcher	Cosinus Similarity

Table 8. User Profile–Item Matching Technique of the Systems based on Content-Based Filtering

2.9.1. Standard Keyword Matching

Standard keyword matching consist in a simple count of the terms which are simultaneously present in the document representation and in the user model [Stefani and Strapparava, 1998]. But, this model has some problems for the synonymy and plural meanings of some words. A lot

of words describe different concepts if used in different contents. For example the words "system", "expert" and "operative": the first and the second word can occur in a document about expert systems, while the first and the third can be found in operative system pages. So the "system" word can have more than one meanings, depending on the context in which is used.

2.9.2. Cosinus Similarity

A early similarity formula was used by Salton in the SMART system [Salton and McGill, 1983]. Salton treated the index and the search query as n-dimensional vectors (see section 2.4.2). The angle between two vectors has even found to be a useful measure of content similarity. The cosine formula calculates the cosine of the angle between the two vectors. As the cosine approaches "1", the two vectors become coincident. If the two vectors are totally unrelated, the they will be orthogonal and the value of the cosine is "0". Moreover, the square of the cosine of that angle (easily computed as the normalized inner product of the two vectors) can be used to rank order the documents. Some approaches have been developed based on this method [Salton and Buckley, 1988], [Buckley et al., 1996], [Yan and Garcia-Molina, 1995], [Chen et al., 2000].

2.9.3. CBR

Retrieval and adaptation techniques from Case-Based Reasoning (CBR) have become very important techniques for realizing intelligent recommendation agents [Cunningham et al, 2001]. The core of such applications is a item database that describes the specific features of each available item. When applying CBR, this item database is treated as a case-base. The case-base contains the old cases in the form of frames, whose slots contain the <document representation, old solution> pairs. The item representation should be its features and the old solution should be it "score" according to a given user model. During the case retrieval phase, item cases are retrieved based on the similarity between the item features and the requirements elicited by the user. The similarity encodes the knowledge to assess whether a item is suitable for the user's interests. Typically, similarity between two cases is calculated through nearest neighbor approaches (see section 2.9.5).

For instance, in the WEBSELL retrieval component [Cunningham et al., 2001], similarity is formalized through similarity measures that are modelled by combining several parametrizable local similarity measures for individual product features with a global aggregation function.

2.9.4. Naive Bayesian Classifier

The naive Bayesian classifier is a probabilistic learning algorithm for classification [Duda and Hart, 1973] based on the Bayes probability formula. The idea is to calculate the probability that a new item belongs to a predefined class. The typical classes to be classified in are interesting and not interesting [Billsus and Pazzani, 1999], but the algorithm can classify items into any set of classes (e.g., relevant, undefined, not relevant). The item must be represented as a feature vector (see section 2.4.2). Therefore each feature indicates the presence, frequency or probability of an attribute in the item (e.g., words). The probability of an item belonging to a specific class is computed as a product of the probabilities of each feature belonging to the class. The feature probability can be easily estimated from training data making the naive assumption that features are independent given the class . Thus, the new item is assigned to the class with the highest probability. Naive Bayes has been shown to perform competitively with more complex algorithms and has become and increasingly popular algorithm in text

classification applications [Pazzani and Billsus, 1997]. Systems that use the naive Bayesian classifier are Personal WebWatcher [Mladenic, 1996], LaboUr [Schwab et al., 2001], News Dude [Billsus and Pazzani, 1999] and Syskill&Webert [Pazzani et al., 1996].

2.9.5. Nearest Neighbor

The nearest neighbor algorithm [Duda and Hart, 1973] operates by storing all examples in the training set. To classify an unseen instance, it assigns it to the class of the most "similar" example. Typically, the similarity measure is associated to a similarity function that calculates the distance between the new item features and the features of the training examples. Depend on the item representation the function can be a simple keyword matching or a weighted comparison [Schwab et al., 2001]. For instance, Syskill & Webert [Pazzani et al., 1996] is implemented with binary features, thus, the most similar example is the one that has the most feature values in common with a test example.

PEBLS [Cost and Salzberg, 1993] is a nearest neighbor algorithm that makes use of a modification of the value difference metric, MVDM, for computing the distance between two examples. This distance between two examples is the sum of the value differences of all attributes of the examples. In many ways, PEBLS is similar to naive Bayesian classifier [Pazzani and Billsus, 1997]. However, PEBLS can accurately learn non-linearly separable concepts from Boolean features while the Bayesian classifier cannot.

2.9.6. Classifiers

Systems based on content-based filtering can handle the recommendation task as a classification task. Based on a set of item features, the system tries to induce a model for each user that allows us to classify unseen items into two or more classes, for example like and dislike (see section 2.7.4). This means that user profile is represented as a classifier: a neural network (see section 2.4.6.1), decision tree (see section 2.4.6.2), inducted rules (see section 2.7.3.1) or a Bayesian network (see section 2.4.6.4).

For instance, Re:Agent [Boone, 1998] implemented a neural network to divide several folders of e-mail into two categories: "work" and "other". Syskill&Webert [Pazzani et al., 1996] used a decision tree to classify Web pages into interesting/not interesting. Recommender [Basu et al., 1998] implemented a rule induction method to classify movies.

2.9.7. Others

A few systems develop their own approaches, typically based on the techniques cited before. For instance, [Morita and Shinoda, 1994] implement the sub-string indexing model or [Chen et al., 2000] propose the TW2 algorithm.

2.10. User profile Matching

Systems based on collaborative filtering match people with similar interests and then make recommendations on this basis (see section 2.3.3). Generally speaking the process of computing a recommendation consist of three steps:

Find similar users

Standard similarity measures are used to compute the distance between the current user's representation and the representation of a set of users. In smaller applications these may be all users; in larger systems statistical sampling methods are used to find a representative subset for which similarity is computed. Sections from 2.10.2 to 2.10.5 show different commonly used techniques to find similar users.

Create a neighborhood

When systems look for similar users, they form a neighborhood of the most similar users to the target user. Generally, two techniques have been used to determine how many neighbors to select: the correlation-thresholding technique and the best-n-neighbors technique. The correlation-thresholding technique is to set an absolute correlation threshold, where all neighbors with absolute correlation greater than given thresholds are selected. Setting a high threshold limits the neighborhood to containing very good correlates, but for many users high correlates are not available, resulting in a small neighborhood that cannot provide prediction coverage for many items. The best-n-neighbors technique is to pick the best-fixed number of users. This technique performs reasonably well, as it does not limit prediction coverage. However, picking a larger number will result in too much noise for those who have high correlates. Picking a smaller number can cause poor predictions for those users who do not have any high correlates. Another approach have been proposed for neighborhood formation by [Herlocker et al., 1999] based on the centroid. The first step is picking the closest user to the target user and calculate the centroid. Then, other users are included in the neighborhood based on the distance to the centroid, which is recalculated each time that a new user is added. Basically, this algorithm allows the nearest neighbors to affect the formation of the neighborhood and it can be beneficial for very sparse data sets.

Compute a prediction based on selected neighbors

The final step is to derive the recommendations from the neighborhood of users. Once the neighborhood has been selected, the ratings from those neighbors are combined to compute a prediction, after possibly scaling the ratings to a common distribution. Different techniques are used in the current systems. The most-frequent item recommendation looks into the neighborhood and for each neighbor scans through the user's interests and extract the most frequently selected items. After all neighbors are accounted for, the system sorts the items according to their frequency and simply returns the n most frequent items as recommendation that have not yet been selected by the active user. The association rule-based recommendation is based on the association rule-based top-n recommendation technique described in section 2.7.3.1. However, instead of using the entire population of users or items to generate rules, this technique only considers the neighborhood generated previously. Note that considering only a few neighbors may not generate strong enough association rules in practice, which as a consequence, may result in insufficient items to recommend. This can be augmented by suing a scheme where the rest of the items, if necessary, are computed by using the most frequent item algorithm Another way to combine all the neighbor's ratings into a prediction is to compute a weighted average of the ratings, using the correlation as the weight. The basic weighted average makes an assumption that all users rate on approximately the same distribution. The approach taken by GroupLens [Resnick et al., 1994] was to compute the average deviation of a neighbor's rating from that neighbor's mean rating, where the mean rating is taken over all items that the neighbor has rated. The justification for this approach is that users may rate distributions centered on different points. An extension to the GroupLens algorithm is to account for the differences in spread between user's rating distributions by converting ratings to z-scores, and computing a weighted average of the z-scores.

Thus, the most important step in systems based on collaborative filtering is computing the similarity between users. But, systems cannot work with large sets of data containing all the users with their features, since the performance of the system will gradually fall down. Therefore, the first part of this section present how to reduce the dimensionality. After this, the common techniques used to compute the similarity between users are explained (the nearest neighbor, clustering and classifiers). Table 9 shows the user profile matching techniques used by the different analyzed systems.

NAME	TECHNIQUE
Anatagonomy	Cosinus Similarity
Beehive	Sharing news among users of the same cluster
Bellcore Video Recommender	Nearest Neighbor (Pearson r Correlation)
Casmir	Pre-Search Request Based Collaboration, Pot-Search Informing
Fab	Cosinus Similarity
GroupLens	Nearest Neighbor (Pearson r Correlation)
Krakatoa Chronicle	Cosinus Similarity
LaboUr	Clustering (Nearest Neigbbour - Pearson r Correlation)
MovieLens	Cosinus Similarity
NewsWeeder	Cosinus Similarity
Personal WebWatcher	Naive Bayesian Classifier
Recommender	Inducted Rule Execution
Ringo / FireFly	Nearest Neighbor (Mean Squared Differences, Pearson r Correlation, Constrained Pearson r Correlation, Artist-Artist)
Smart Radio	Nearest Neighbor (Pearson r Correlation)
Tapestry	Tapestry Query Language
WebSell	CBR with Nearest Neighbor (Pearson r Correlation)
Websift	Rule Execution and Pattern Matching
WebWatcher	Cosinus Similarity

Table 9. User Profile Matching Technique of the Systems based on Collaborative Filtering

2.10.1. Dimensionality Reduction

For large databases containing many users we will end up with thousands of features. Working under these conditions is not practical, because the amount of data points needed to approximate a concept in *d* dimensions grows exponentially with *d*. This is, of course, not a problem unique to collaborative filtering (see section 2.7.2.1). Essentially, this approach takes the user-item ratings matrix (see section 2.4.7) and uses any technique to obtain a reduced matrix.

Researchers in information retrieval have proposed different solutions to the text version of this problem. One of these approaches, Latent Semantic Indexing (see section 2.7.2.1.5) is based on dimensionality reduction of the initial data through singular value decomposition (SVD). In the same way, this technique can also be used in collaborative filtering systems to reduce the user-item ratings matrix [Billsus and Pazzani, 1998].

[Hofmann and Puzicha, 1999] propose two latent class models for the same purpose. The aspect model is a probabilistic latent space model which models individual preferences a convex combination of preference factors (most appropriate for prediction and recommendation). The two-side clustering model simultaneously partitions persons and objects into clusters (most appropriate for identifying meaningful groups or clusters).

[Hayes et al., 2001] propose the Case Retrieval Nets (CRN) for systems that apply case-based reasoning techniques (see section 2.9.3) to the collaborative filtering. A CRN is a memory model that builds a net instead of a tree from the case base. It uses organizational features derived from associative memory structures and spreading activation process similar to that

used in connectionist models. The main benefit is that new cases and case features can be added without having to rebuild the memory structure, the principal shortcoming of the case-trees.

The reduced representation of the user-item ratings matrix has several advantages:

- First, it alleviates the sparsity problem (see section 2.3.3) as all the entries in the reduced matrix are nonzero, which means that all the users now have their opinions on the items.

- Second, the scalability problem (see section 2.3.3) also is almost solved since both the processing time and storage requirement improve dramatically.

- Third, this reduced representation captures latent association between users and items in the reduced feature space and thus can potentially remove the problem of synonym words.

- Fourth, the reduced representation contributes to improve the performance of the system [Billsus and Pazzani, 1998].

2.10.2. Nearest Neighbor

Nearest neighbor algorithms are based on computing the distance between consumers based on their preference history. Predictions of how much a user will like a item are computed by taking the weighted average of he opinions of a set of nearest neighbors for that product. Neighbors who have expressed no opinion on the product in question are ignored. Nearest neighbor algorithms have the advantage of being able to rapidly incorporate the most up-to-date information, but the search for neighbors is slow in large databases.

[Herlocker et al., 1999] compare different nearest neighbor techniques and show as conclusions the results of these techniques in a specific framework and the suitability of each one in different recommendation systems.

Mainly, two approaches are used in current systems to calculate the similarity between users:

2.10.2.1. Cosinus Similarity

One of the easiest ways to compute the similarity between an item and a user in user profile-item matching techniques is to represent items and profiles as vectors (see section 2.4.2) and computing the cosine of the angle formed by the two vectors (see section 2.9.2). The same formalism can be adopted to collaborative filtering, where users are compared to other users in the same way. The vector similarity measure has been shown to be successful in information retrieval [Salton and McGill, 1983]. However [Breese et al., 1998] has found that vector similarity does not perform as well as Pearson correlation (see section 2.10.2.2) in Collaborative Filtering systems.

2.10.2.2. Correlation

Working with databases of user ratings for items, where users indicate their interest in an item on a numeric scale, it is easy to define similarity measures between two user profiles based on the correlation between the users.

A correlation measure proposed by [Shardanand and Maes, 1995] is the Pearson correlation coefficient. Pearson correlation measures the degree to which a linear relationship exists between two variables. It is derived from a linear regression model that relies on a set of

assumptions regarding the data, namely that the relationship must be linear, and the error must be independent and have a probability distribution with mean 0 and constant variance for every setting of the independent variable. Thus, this coefficient ranges from –1 (indicating a negative correlation), via 0 (indicating no correlation) to +1 (indicating a positive correlation between two users). In contrast with other algorithms, this algorithm makes use of negative correlation as well as positive correlation to make predictions.

Spearman rank correlation coefficient [Herlocker et al., 1999] is similar to Pearson, but does not rely on model assumptions, computing a measure of correlation between ranks instead of ratings values. Spearman correlation performed as well as Pearson correlation and because it is not dependent on model assumptions, it should perform consistently across diverse datasets.

These correlation-based prediction schemes were shown to perform well, but they suffer from several limitations [Billsus and Pazzani, 1998]:

- First, correlation between two user profiles can only be computed based on items that both users have rated (i.e., the summations or averages). If users can choose among thousands of items to rate, it is likely that overlap of rated items between two users will be small in may cases. Therefore, many of the computed correlation coefficients are based on just a few observations, and thus the computed correlation cannot be regarded as a reliable measure of similarity. For example, a correlation coefficient based on three observations has as much influence on the final prediction as a coefficient based on 30 observations.

- Second, the correlation approach induces one global model of similarities between users, rather than separate models for classes of ratings (e.g., positive ratings vs. negative ratings). Current approaches measure whether two user profiles are positively correlated, not correlated at all or negatively correlated. However, ratings given by one user can still be good predictors for ratings of another user, even if the two user profiles are not correlated.

- Third, an maybe most importantly, two users can only be similar if there is overlap among the rated items, i.e., if users did not rate any common items, their user profiles cannot be correlated. Due to the enormous number of items available to rate in many domains, this seems to be a serious stumbling block for many filtering services, especially during the startup phase. However, just knowing that users did not rate the same items does not necessarily mean that they are not like-minded. We believe that potentially useful information is lost if this kind of transitive similarity relation cannot be detected.

2.10.2.3. Others

Another approach based on correlation between users is the entropy-based uncertainty measure. The measure of association based on entropy uses conditional probability techniques to measure the reduction in entropy of the active user's ratings that results from knowing the another user's ratings. [Herlocker et al., 1999] exhibit that entropy has not shown itself to perform as well as Pearson correlation. [Shardanand and Maes, 1995] a part of Pearson r Correlation and Constrained Pearson r Correlation use the Mean Squared Differences algorithm, which perform well compared to Pearson correlation. Another more complicated approach is explained in [Greening, 1997].

2.10.3. Clustering

Earlier, the user modeling community provided a different answer, namely the stereotype approach [Rich, 1979]. During the development time of a system, user subgroups are identified and typical characteristics of members of these subgroups determined. During the runtime of the

system, user is assigned to one or more of these predefined user groups and their characteristics attributed to the user. The need for an (empirically based) pre-definition of these stereotypes is an evident disadvantage. As an alternative, the system Doppelganger used clustering mechanisms to find user groups dynamically, based on all available individual user models [Orwant, 1995]. Explicitly represented user models can be clustered and the descriptions of the clusters can be used like predefined stereotypes. Doppelganger compensates for missing or inaccurate information about a user by using default inferences from communities, which resemble traditional user modeling stereotypes with two major differences: membership is a not all-or-nothing, but a matter of degree; and the community models are computed as weighted combinations of their member user models, and thus change dynamically as the user models are augmented. Once the clusters are created, predictions for an individual can be made by averaging the opinions of the other users in that cluster.

Some clustering techniques represent each user with partial participation in several clusters. The prediction is then an average across the clusters, weighted by degree of participation. Clustering techniques usually produce less-personal recommendations than other methods, and in some cases, the clusters have worse accuracy than nearest neighbor algorithms [Breese et al., 1998]. Once the clustering is complete, however, performance can be very good, since the size of the group that must be analyzed is much smaller.

In contrast to real stereotypes, clusters are acquired dynamically and can be revised whenever needed. Thus dynamic evolution of user groups can be accounted for.

2.10.4. Classifiers

Collaborative filtering can be seen as a classification task [Billsus and Pazzani, 1998]. Based on a set of ratings from users for items, we try to induce a model for each user that allows us to classify unseen items into two or more classes, for example like and dislike (see section 2.7.4). Alternatively, if the goal is to predict user ratings on a continuous scale, the system has to solve a regression problem. Typically, the initial data exists in the form of a sparse matrix (see section 2.4.7), where rows correspond to users, columns correspond to items and the matrix entries are ratings. Note that sparse in this context means that most elements of the matrix are empty, because every user typically rates only a very small subset of all possible items. The prediction task can now be seen as filling in the missing matrix values. Since we are interested in learning personalized models for each user, we associate one classifier with every user. This model can be used to predict the missing values for one row in our matrix.

[Basu et al, 1998] built a hybrid recommender system that mixes collaborative and content filtering using an induction learning classifier. [Good et al., 1999] implemented induction-learned feature-vector classification of movies and compared the classification with nearest neighbor, but that combining the two added value over nearest neighbor alone. [Billsus and Pazzani, 1998] format the data set of user ratings in the vector space model and then, they use a neural network to predict the missing values. [Breese et al., 1998] results indicate that for a wide range of conditions, Bayesian networks with decision trees at each node outperform the other approaches.

2.10.5. Others

Another approach called Horting was proposed by [Wolf et al., 1999]: Horting is a graph-based technique in which nodes are users, and edges between nodes indicate degree of similarity between two consumers. Predictions are produced by walking the graph to nearby nodes and combining the opinions of the nearby consumers. Horting differs from nearest neighbor as the

graph may be walked through other users who have not rated the product in question, thus exploring transitive relationships that nearest neighbor algorithms do not consider. In one study using synthetic data, Horting produced better predictions than a nearest neighbor algorithm.

2.11. Evaluation of the System

Unfortunately, only a few systems evaluate and discuss their results scientifically. This is in part due to the fact that it is actually hard to determine how well a personalization systems works, as this involves purely subjective assessments. However, some approaches are discussed in this section, but due to a lack of data, a comparison of the different systems with respect to performance is currently impossible.

Table 10 shows the evaluation system techniques used by the different analyzed systems.

NAME	TECHNIQUE
ACR News	Logs
Amazon	Real
Amalthaea	Logs - Fitness
Anatagonomy	Logs - Correlation
Beehive	Nothing
Bellcore Video Recommender	Logs – Accuracy (Correlation)
Casmir	User Simulator - Precision
CDNow	Real
Fab	Evaluation - Ndpm
GroupLens	Not Specified
ifWeb	Evaluation – Precision, Ndpm
InfoFinder	Not Specified
INFOrmer	Nothing
Krakatoa Chronicle	Nothing
LaboUr	Logs - Accuracy
Let's Browse	Evaluation
Letizia	Nothing
LifeStyle Finder	Evaluation
MovieLens	Logs – Accuracy (MAE, ROC)
News Dude	Logs – Accuracy, F-mesure
NewsWeeder	Not Specified
NewT	Evaluation, User Simulator – Precision, Recall
Personal WebWatcher	Logs – Precision, Accuracy
PSUN	Nothing
Re:Agent	Logs - Precision
Recommender	Logs - Precision, Recall
Ringo / FireFly	Logs – Accuracy (MAE)
SIFT Netnews	Nothing
SiteIF	Nothing
Smart Radio	Nothing
Syskill & Webert	Logs - Accuracy
Tapestry	Nothing
Webmate	Logs - Accuracy
WebSail	Logs - Recall
WebSell	Nothing
Websift	Logs
WebWatcher	Evaluation - Accuracy

Table 10. Evaluation Technique of the systems.

This section is organized in two different parts. The first one shows several methods to acquire results and the second one shows metrics to evaluate these results.

2.11.1. Results Acquisition

The acquisition of results is a critical task in the evaluation of the systems. Current systems use one of the following approaches: a real environment, an evaluation environment, current logs of the system or a user simulator.

2.11.1.1. Real Environment

The best way to evaluate a personalized system is showing real results obtained in a real environment. Only a few commercial systems like Amazon.com [Amazon] or CDNow.com [CDNow] can show real results based on the economic effect.

2.11.1.2. Evaluation Environment

Some systems are evaluated in the laboratory letting a set of users interact with the system during a period of time. Usually, the results are not enough reliable because the users know the system or the purpose of the evaluation. A original approach was accomplished by NewT [Sheth, 1994]; in addition to the numerical data collected in the evaluation sessions, a questionnaire was also distributed to the users to get feedback on the subjective aspects of the system.

2.11.1.3. Logs

Most of the systems are evaluated analyzing or validating the logs. A commonly used technique is the "10-fold cross-validation technique". It consists in validate the logs predicting the relevance (e.g., ratings) of the recorded examples (see Figure 13). Then, the guessed ratings are compared to the ratings of the logs.

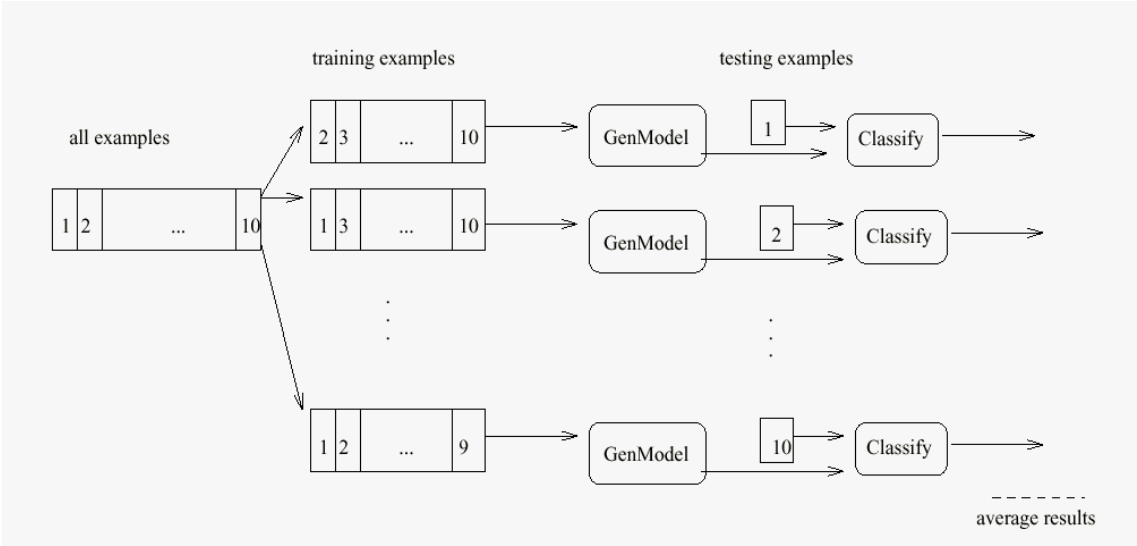

Figure 13. "10-fold cross-validation technique" [Mladenic, 1996]

2.11.1.4. User Simulator

Important issues such as the learning rates and variability in learning behavior across large heterogeneous populations could be investigated with large collections of simulated users whose design was tailored to explore those issues. This enables large-scale experiments to be carried out quickly and also guarantees that experiments are repeatable and perfectly controlled. This also allows researchers to focus on and study the behavior of each sub-component of the system, which would otherwise be impossible in an unconstrained environment. For instance, [Holte and Yan, 1996] conducted the experiments using an automated user called Rover that played the role of the user, rather than human users. [Sheth and Maes, 1993] and [Berney and Ferneley, 1999] also used a user simulator to evaluate the performance of the systems.

2.11.2. Results Evaluation

Once the results are available, we need to evaluate them. A set of metrics is proposed for this purpose: coverage, recall, precision, f-measure, fallout, NDPM and accuracy.

2.11.2.1. Coverage

Coverage is a measure of the percentage of items for which a recommendation system can provide predictions. A low coverage value indicates that the user must either forsake a large number of items or evaluate them based on criteria other than recommendations. A high coverage value indicates that the recommendation system provides assistance in selecting most of the items.

2.11.2.2. Recall

The Recall measure [Salton and McGill, 1983] is the fraction of the actual set of relevant items that are correctly classified as relevant. It's a measure of selection effectiveness and represents the probability that a relevant document will be selected.

2.11.2.3. Precision

The Precision measure [Salton and McGill, 1983] is the fraction of the selected items which are actually relevant to the user's information need. It's also a measure of selection effectiveness and represents the probability that a selected item is relevant.

2.11.2.4. F-Measure

Sometimes it is important to evaluate precision and recall in conjunction, because it is easy to optimize either one separately. The F-Measure [Lewis and Gale, 1994] consists in a weighted combination of precision and recall that produces scores ranging from 0 to 1.

2.11.2.5. Fallout

The Fallout measure [Salton and McGill, 1983] is the fraction of the non-relevant items that are selected. It's a measure of rejection effectiveness.

2.11.2.6. NDPM Measure

The Normalized Distance-Based Performance Measure (NDPM) [Yao,1995] is a measure of the capability to order correctly the items from interesting to not-interesting. Yao developed NDPM theoretically, using an approach from decision and measurement theory. User ratings could be transformed to binary ratings (if they were not already), and NDPM could be used to compare the results to the system ranking. One of the key weaknesses of NDPM with respect to evaluating ranked retrieval is the lack of a statistical significance test.

2.11.2.7. Accuracy

Typically, the accuracy metric is defined as the percent of correctly classified items. For instance, the number of interesting news articles divide by the total number of news articles in a newspaper. However, [Sarwar et al., 1998] gather and classify from prior research different ways to measure it:

- Statistical Recommendation Accuracy: measures the closeness between the numerical recommendations provided by the system and the numerical ratings entered by the user for the same items. Three versions of this measure are used:

  - CORRELATION is a statistical measure of agreement between two vectors of data, typically between ratings and predictions. Pearson correlation coefficient is the most commonly used. A higher correlation value indicates more accurate recommendations.

  - MAE - MEAN ABSOLUTE ERROR is a measure of the deviation of recommendations from their true user-specified values. The lower MAE, the more accurately the recommendation engine predicts user ratings.

  - RMSE - ROOT MEAN SQUARED ERROR is a measure of error that is biased to weigh large errors disproportionately more heavily than small error. Lower RMSE indicates better accuracy.

- Decision-Support Accuracy: measures how effectively recommendations help a user select high-quality items. Three versions of this measure are used :

  - REVERSAL RATE is a measure of how often the system makes big mistakes that might undermine the confidence that a user has in the recommendation systems. Low reversals refer to cases where the user strongly dislikes an item that the system strongly recommends. High reversals are cases where the user strongly likes the item, but the system recommendation is poor.

  - ROC SENSITIVITY is a measure of the diagnostic power of a filtering system. Operationally, it is the area under the receiver operating characteristic (ROC) curve, a curve that plots the sensitivity and specificity of the test. Sensitivity refers to the probability of a randomly selected good item being accepted by the filter. Specificity is the probability of a randomly selected bad item being rejected by the filter. Therefore, the ROC sensitivity measure is an indication of how effectively the system can steer people towards high-rated items and away from low-rated ones.

  - PRC SENSITIVITY is a measure of the degree to which the system presents relevant information. Operationally, it is the area under the precision-recall curve (PRC). Precision measures the percentage of selected documents that are relevant; recall measures the percentage of relevant documents that are selected. Hence, precision

indicates how selective the system is and recall indicates how thorough it is in finding valuable information. A higher value is more accurate.

2.12. System Architecture

For simplicity purposes, in the whole paper, the general word "system" is used to mention the current personalized applications. However, some applications are structured as either intelligent agents or ecosystems of agents. Therefore, they can be mentioned as personalized agents or personalized ecosystems of agents.

2.12.1. Agent

There is no clear definition for the term agent, but the following two definitions (one general and the second one closer to this work) are largely accepted by the researchers:

[Wooldridge, 1999]: "An agent is a computer program that is situated in some environment, and that is capable of autonomous acting in this environment in order to meet its design principles".

This definition can be extended to define which are the conditions for calling an agent an intelligent agent [Wooldridge and Jennings, 1995]:

Autonomy: operates without the direct intervention of humans or other agents, and has control over its actions and internal state.

Reactivity: intelligent agents are able to perceive their environment, and respond in a timely fashion to changes that occur in it in order to satisfy their design principles.

- Pro-activeness: intelligent agents do not simply act in response to their environment, they are able to exhibit opportunistic, goal-directed behavior by taking the initiative where appropriate.

- Social ability: intelligent agents are capable of interacting with other agents (and possibly humans) in order to satisfy its design principles. One of the most important aspects in agents is social ability, social ability can be understood [Wooldridge, 1999] as the necessity to negotiate and co-operate with other to achieve goals.

Taking this paradigm, autonomous agents can be developed which co-operate with each other. Every agent represents an unique user and they operate as a personal assistant, for instance, guiding the user in the query formulation process, storing and managing the user's spheres of interest and pro-actively recommending items that may be of interest to the user.

2.12.2. Ecosystem of Agents

Ecosystems are complex biological systems in which adaptation is an essential characteristic [Devine et al., 1997]. Some mathematical models of ecosystems simulate models of heterogeneous agents that evolve in a system, according to their fitness to some aspect of the ecosystem. Normally these agents compete for resources. The most successful species tend to create new ones, combining their own information and adding new one through Genetic Algorithm (GA) [Mitchell, 1996] or other similar techniques [Mitchell, 2000].

Building on this idea, [Sheth and Maes, 1993] implemented an ecosystem architecture of agents to filter Internet News in a system called 'Newt'. A genetic algorithm uses algorithmic analogues to the genetic crossover and mutation operations to generate candidate profiles that inherit useful features from their ancestors, and uses competition to identify and retain the best ones. The crossover operator was periodically applied to combine segments of two candidate profiles which were among those that had produced the highest ranks (using a cosine similarity measure) for articles that the user later identified as desirable. A mutation operator was sometimes applied to the newsgroup name to explore whether existing candidate profiles would perform well on newsgroups with similar names. All of the candidate profiles contributed to the ranking of the documents shown to the user, although those, which consistently performed well, contributed more strongly to the ranking. Hence, the profile itself was determined by the population of candidate profiles, rather than by any individual candidate.

A similar approach was implemented in Amalthaea [Moukas, 1997] by creating an artificial ecosystem of evolving agents that cooperate and compete in a bounded resource environment. New agents are created by crossover or mutation (or both). Both operators are applied to the evolvable part of the agents, the genotype. The other part of the agents, the phenotype contains information that should not be evolved, usually instructions on how to handle the evolvable part. The two point crossover operator works as follows: given two agents returns two new agents that inherit a part of the keyword vectors of the parents. The operator randomly selects two points in the keyword vector and exchanges all the fields of the two parents that lie between these points, creating two new agents. Mutation is another method for creating offspring agents. The mutation operator takes the genotype of an agent as argument and creates a new agent that is a randomly modified version of its parent. The weights of the mutated keywords are modified randomly while the new mutated keyword is a randomly selected keyword from an agent that belongs to another cluster.

The Fab [Balabanovic and Shoham, 1997] and PSUN [Sorensen and McElligot, 1995] systems also implemented this architecture.

2.13. Conclusions

With the unceasing growing of the Internet and its environment, the necessity of a new technology, which assists users to find their objectives, comes up. The combination of modeling of particular user preferences, building content models and modeling of social patterns in intelligent agents seems to be a charming solution. The current state of the art in personalized systems on the Internet is analyzed to draw a general taxonomy. The taxonomy is, at first, classified in two main groups: user profile generation and maintenance, and user profile exploitation. Then, under this general classification, 10 common features are extracted and inside each feature, all the used techniques for the analyzed systems are briefly explained. There is no intention to give a guide for the researchers to implement their own systems, the intention is to give the current state of the art organized in a simple classification, explaining the used methods and in some cases exhibit their advantages and disadvantages. Thus, the main purpose is to give a stating point for the researchers to construct their own personalized system.

3. THE PROPOSAL

As seeing in the state of the art, learning is a key issue in personalized agents due to the interactivity and dynamism of the environment (Internet) and the changing preferences of the user. Learning can be accomplished in an isolated way or in a collaborative way. Isolated learning just takes into account the gathered information about the user to induce his preferences. Collaborative learning takes advantage of the community of personal agents, which have the same tasks and objectives, to learn about the user preferences. While isolated learning methods have been explored for long traditional machine learning techniques, collaborative learning investigation is in its beginning. Moreover, the synergy that can be created between both learning methods is a novel research issue. Our proposal focus precisely in the use of both methods of learning.

We propose a personalized agent that handles the learning in both isolated and collaborative ways. Case-Based Reasoning (CBR) is proposed to learn about the user preferences in an isolated way. The maintainability and transparency of CBR systems insure the success in the majority of research fields. Techniques based on trust are proposed to learn in a collaborative way. Based on the agent's theory, we are going to create a community of personal agents that rely in other agents opinion and its own interests to improve their performance. To illustrate the agents performance, we propose the development of an application in the recommender field.

The rest of the proposal is structured as follows. Section 3.1 introduces the architecture of the personalized agent. Section 3.2 presents the CBR approach to personalization. The trust in the collaborative world applied to personalized agents is detailed in section 3.3. Finally, in section 3.4 some conclusions about the proposal are given.

3.1. The Personalized Agent

One of the problems of the user's profile matching techniques analyzed in the state of the art is that the systems need all the information of the different users centralized in the server to compare them. Thus, the system can access the personal information of the users and the privacy issues are not guarantied. In an attempt to guaranty privacy we propose a personalized agent that does not need to be placed in the server (see Figure 14). Thus, the personal information of the users is distributed in their own agents and the only way to know something about the user is with the interaction. We distinguish two levels of interaction: interaction with other agents (peer to peer) and interaction with the server. In the first one, the personal agents look for similar users interacting with the other personal agents ("playing agents"). The interaction consists in "speak" about items, that is, an agent asks for the opinion of different items to the other agents. Different users are similar when their personal agents have a similar opinion about the same items.

The personal agents interact with the server agent to know information about new items or new agents in the system. The server agent interacts with the personal agents to gather information about some items (e.g., how many people is interested in one item) or to communicate the presence of new items or new agents in the system. Therefore, the personal agents and the server agent do not know specific information about the other users (e.g., visited restaurants or purchased products). In this way, the system guaranties the privacy of the user's personal information. Moreover, the personal agents are anonymous for the other personal agents and the

server agent. Each personal agent is identified by an unique id and the other agents just know the id of the other agents and how to contact them. Thus, the agents have no idea who is the user behind the personal agent.

Another important feature is the scalability of the system. When a new user is incorporated to the system, a personal agent is created and its id and contact address is added to the agent's list of the server. As from the interaction of this user with the server agent and the other personal agents, the new user is quickly adapted to the environment. When a new item is incorporated to the system, it is just added to the item database of the server. The server agent communicates the presence of a new item and the personal agents analyze whether it is interesting for their users.

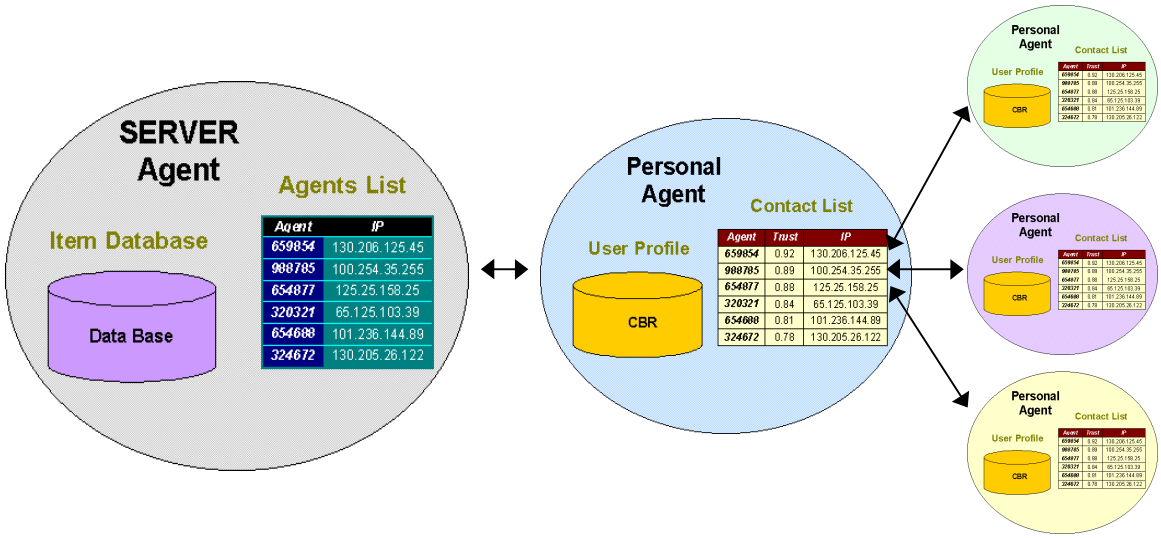

Figure 14. System Architecture

This is an open architecture. The system does not depend on the agent's technology or on the agent's computer host. The system can incorporate anytime an agent that obeys the communication protocols and that can be contacted by another agent in a port of an IP address. It does not matter the programming language of the agent. Moreover, because of this architecture we can test different kinds of agents and find out their performance.

The system is also robust due to its distributed architecture. When the computer of a personal agent falls down, the system is not affected, since the only problem is that the other agents cannot interact with this agent. The problem is bigger when the server falls down, but not critique. The other agents cannot access to the server, thus, they do not know if new items or new agents are incorporated to the system and they do not know if an agent changes its contact address. But, personal agents can go on its work interacting with other personal agents and recommending items.

The system architecture can be classified within multi-agent systems (MAS) in the following way according to [Stone and Veloso, 1997]:

- Decentralized architecture: there is no a central authority, which is in charge of making decisions respecting the composition of co-ordination objects. In this case, self-interested individuals compose the population.

- Homogeneous agents: all of the agents have the same internal structure including goals, domain knowledge, and possible actions. They also have the same procedure for selecting

among their actions. The only differences among agents are their sensory inputs and the actual actions they take: they are situated differently in the world.

- Deliberative agents: agents maintain an internal state trying to adapt their behavior to the user preferences.

- Agents with a local perspective: each agent sees a partial picture of the world. Agents can see their user and the other agents, but they cannot see the other users.

3.2. A Case-Based Reasoning Approach

In the real world, the salesman of the grocery where you always buy your meal knows so much about you. Based on the typical products that you normally buy, he recommends you new items that he thinks you could like. For instance, if you always buy a trademark of milk, the salesman can recommend you, with a high probability of success, another similar one that during this week has a discount. Another typical situation is due to the changing interests of the customers over time. If you do not buy a given product during a long time, the salesman gradually stops recommend you similar products. He guesses that you are not interested in it anymore. The recommendations of the salesman are always based on the knowledge about the products and the expertise about your tastes, preferences, interests and behavior in the grocery. In summary, the experience on your activity on the grocery guides the salesman to improve his sales.

Precisely, learning and reasoning on past experiences is the essentials of CBR. CBR is a recent paradigm to problem solving and learning that has drawn a lot of attention over the last years. Instead of relying just on general knowledge of a problem domain, or making associations along generalized relationships between problem descriptors and conclusions, CBR is able to utilize the specific knowledge of previously experienced, concrete problem situations (cases) [Aamodt and Plaza, 1994]. CBR is based on people reasoning. Humans are robust problem-solvers; they routinely solve hard problems despite limited and uncertain knowledge, and their performance improves with experience. All of these qualities are desirable for real-world AI systems. Consequently, it is natural to ask how CBR can advance AI technology.

[Leake, 1996] identifies five main advantages of CBR over other AI techniques:

- Knowledge acquisition: in many domains, the cost of knowledge acquisition for CBR is very low, because case-based reasoners reason from complete specific episodes and makes unnecessary to decompose experiences and generalize their parts into rules.

- Knowledge maintenance: CBR systems do incremental learning, since the knowledge is easily increased just adding new cases in the case base. Moreover, a user may be able to add missing cases to the case base without expert intervention.

- Increasing problem-solving efficiency: Reuse of prior solutions helps to increase problem-solving efficiency by building on prior reasoning rather than repeating prior effort. In addition, because CBR saves failed solutions as well as successes, it can advise about avoiding potential problems.

- Increasing quality of solutions: When the principles of a domain are not well understood, rules will be imperfect. In that situation, the solutions suggested by cases may be more accurate than those suggested by chains of rules, because cases reflect what really happens (or fails to happen) in a given set of circumstances.

- User acceptance: this is a key issue deploying AI technology: no system is useful unless its users accept its results. Users accept reasoning that seems natural to them.

An added value regarding user acceptance is the transparency of the CBR systems. Black boxes like neural networks cannot provide explanations of their decisions. Rule-based systems must explain their decisions by reference to their rules, which the user may not fully understand or accept. On the other hand, the results of CBR systems are based on actual prior cases that can be presented to the user to provide compelling support for the system's conclusions. Moreover, avoiding generalizations facilitates to take exceptions into account. CBR can model local phenomena well compared to eager techniques that tend to focus on more global models.

However, an important problem comes up when applying CBR: the dimensionality problem. Adding new cases to enlarge the knowledge of the system provokes an uncontrolled growth of the case base. Therefore, the performance of the system gradually decreases for instance due to the increased cost in accessing memory.

Based on the good results that CBR has shown in learning systems [Watson, 1997], our idea is mapping these results to the personalization field. We are not concerned in whether it is a better approach than the others are, we just want to show CBR as a good option in personalized agents. Moreover, in order to cope with the dimensionality problem, our CBR approach includes a forgetting mechanism based on what we call the drift attribute.

In the rest of the section we explain the general features of CBR, emphasizing on the approach we propose.

3.2.1. CBR Methodology

The main idea of CBR is to solve a new problem by retrieving a previous similar situation and by reusing information and knowledge of that situation [Aamodt and Plaza, 1994]. A new problem is solved by finding a similar past case and reusing it in the new problem situation.

An important feature of case-based reasoning is its coupling to learning. The notion of this method does not only denote a particular reasoning method, irrespective of how the cases are acquired, it also denotes a machine learning paradigm that enables sustained learning by updating the case base after a problem has been solved. When a problem is successfully solved, the experience is retained in order to solve similar problems in the future. When an attempt to solve a problem fails, the reason for the failure is identified and remembered in order to avoid the same mistake in the future. Case-based reasoning makes for learning from experience, since it is usually easier to learn by retaining a concrete problem solving experience than to generalize from it.

In general, a new problem is solved by retrieving one or more previously experienced cases, reusing the case in one way or another, revising the solution based on reusing a previous case, and retaining the new experience by incorporating it into the existing case base.

Thus, a general CBR cycle (see Figure 15) may be described by the following four processes:

1. RETRIEVE the most similar cases
2. REUSE the information and knowledge in that case to solve the problem
3. REVISE the proposed solution
4. RETAIN the parts of this experience likely to be useful for future problem solving.

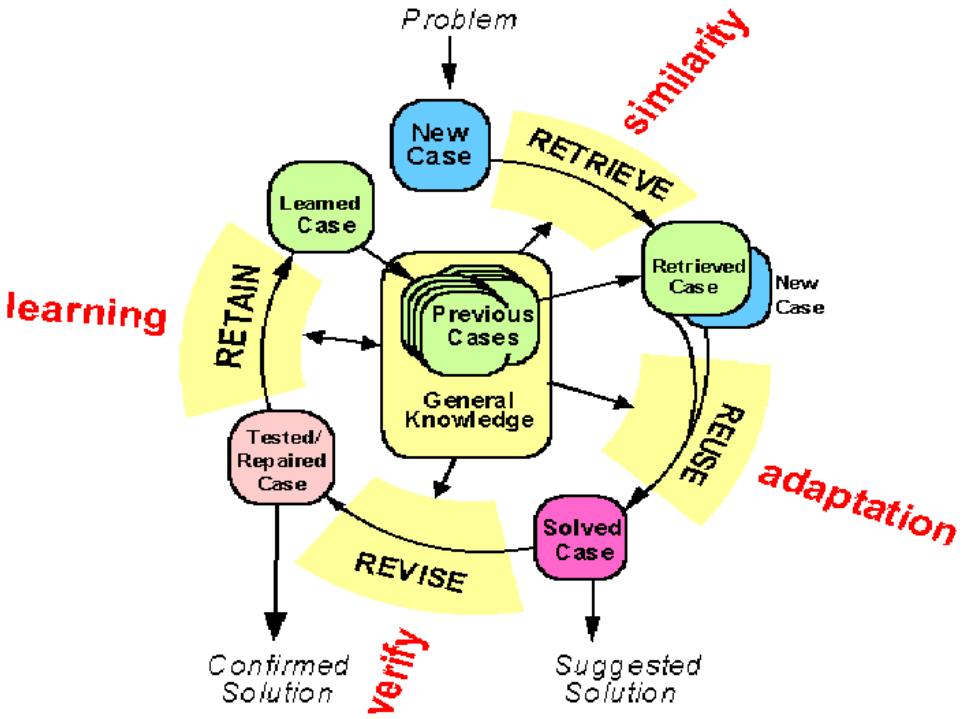

Figure 15. CBR Cycle [Aamodt and Plaza, 1994]

The core of CBR is the concept similarity used to retrieve similar situations. Obtaining a similarity value between two different items is a useful idea that can help us to improve our systems. However, similarity is not a simple or uniform concept. Similarity is a subjective term that depends on what one's goals are. For instance, a shoe is similar to a hammer if one is looking around for something to bang with, but not if one wants to extract nails. This is a silly example but the idea is applicable to any field: two products with the same price would get the maximum similarity if the user is interested in restaurants with the same price, but may differ greatly on another goal, such as quality or trademark.

3.2.2. Overview of the CBR Approach to Personalization

In CBR terminology, a case usually denotes a problem situation, a previously experienced situation, which has been captured and learned in a way that it can be reused in the solving of future problems. The case has two parts, the problem definition and the solution of the problem. When we apply CBR to personalization the case becomes a previous item that can give us information about the interest of the user about it. Then, the case has also two parts, the item definition (the problem) and the interest that the user has shown about it (the solution).

To evaluate if a new item can be interesting to the user, the agent searches into the case base similar items. If the interest value is high enough, the item is recommended to the user.

In order to learn, CBR should have some feedback about the usefulness of its recommendation. Then, the system looks over the interaction of the user with the new item in order to know whether the user is really interested in the new item. If this is the case, the new item is inserted to the case base with the interest attributes.

Regarding the CBR cycle:

- In the retrieve phase, as from a new item, the system searches similar items in the case base in order to know whether the user can be interested in it. According to the previous case representation, two similar items will correspond to a same interest.

- In the reuse phase, as from the retrieved set of similar items, the system calculates a confidence value of interest to recommend the new item to the user.

- In the revise phase, as from the relevance feedback of the user, the system evaluates the interest of the user about the new item. The idea is tracking the user interaction with the system to know relevant information about the user interest on the recommended item.

- In the retain phase, the new item is inserted in the case base with the interest attributes added in the revise phase. This information will be available in future recommendations to know the user interests about similar items.

In the following sections the structure of the case base, the different CBR phases of the new approach and the initial case base generation are explained.

3.2.3. The Case Base

A case-based reasoner is heavily dependent on the structure/representation and content of its collection of cases. Moreover, the dimension of the case base is also an open problem, since it affects the performance of the system. Following, we introduce the case base representation we will use for personalization and how we solve the dimensionality problem with a control attribute called the drift attribute.

3.2.3.1. The Case Base Representation

The representation problem in CBR is primarily the problem of deciding what to store in a case, finding an appropriate structure for describing case contents, and deciding how the case memory should be organized and indexed for effective retrieval and reuse.

In our approach a case is split into two part: a first set of attributes describing the item (the definition of the problem in CBR terminology) and a second set of attributes describing the interest of the user (the solution of the problem in CBR terminology). We decided that the agent just handles items from one topic (e.g., restaurants). In such a way, we simplify the case base of the personal agent, since items from the same topic have the same attributes. As shown in Figure 16 from the restaurants recommendation domain, the case representation consists in a set of item attributes describing the restaurant and a set of interest attributes describing the opinion of the user about the restaurant.

Using CBR to look for similar items is not an original approach. WebSell [Cunningham, 2000] recommends products in the e-commerce field based on the CBR concepts, and Entree [Burke et al., 1997] uses CBR to recommend restaurants. However, these systems just keep a list of items that the user likes or dislikes. The items of the list constitute the case base and they are represented as a set of attributes. The main difference with our approach is that we keep also a set of interest attributes as a solution of the case, that is, an explicit representation of user interests.

The system has not expert knowledge about the domain of recommendations. Thus, the success of the system depends only on the description of the items. To create a attribute set for an item,

we must make use of whatever information is available about the item's qualities. Item descriptions in general do not tend to be very complex, consisting largely of descriptive adjectives, nouns or values. For example, when the CBR goal is to recommend restaurants, the system can deal with features of capacity (e.g., "100 places" or "150 places"), qualities of the cuisine (e.g., "traditional", "creative" or "bland") or approximated price (e.g., "from $10 to $15" or "from $20 to $30"). The item attributes are structured like a typical relational database.

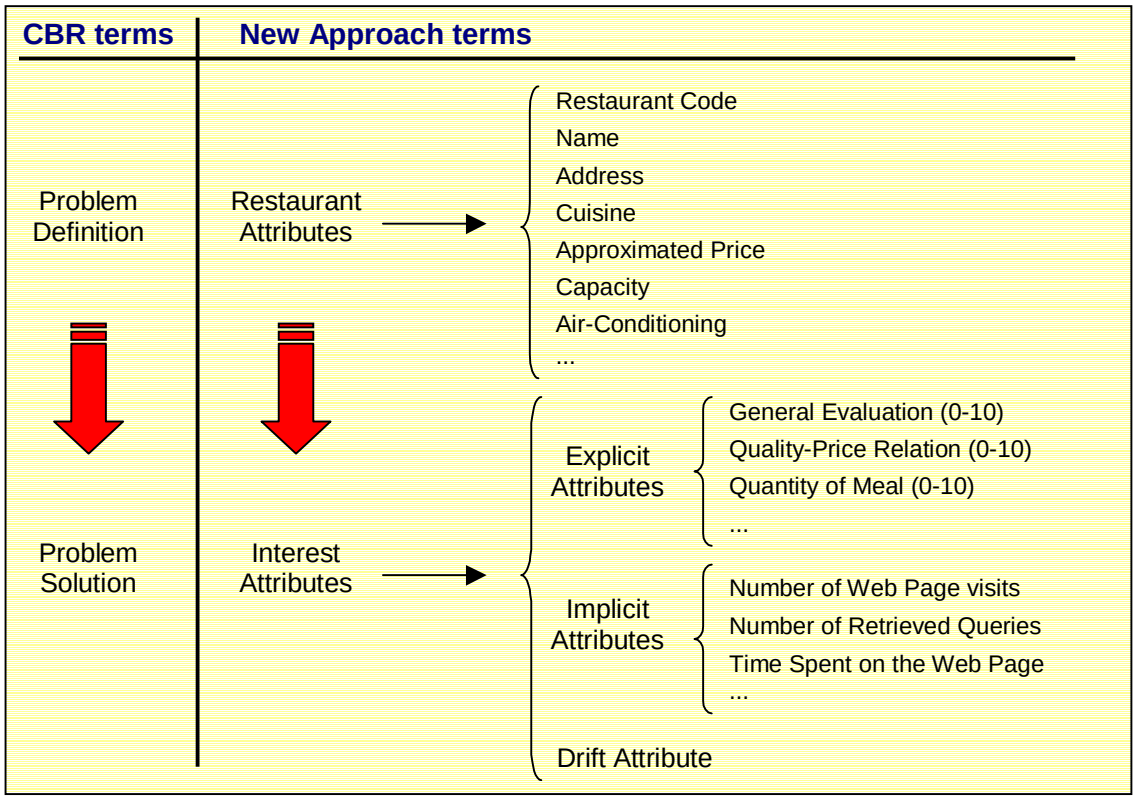

Figure 16. An Example of Case Representation in the Restaurants Domain

The interest attributes keep all the information the agent gathers from the user. These attributes depend on the technique to get relevance feedback: explicit or implicit (see section 2.6). The explicit feedback relies on the fact that any information can be asked to the user explicitly, such as quality-price relation or quantity of meal. On the contrary, implicit information can be captured from the interaction between the user and the system like consulted items, time spent consulting items, the number of visits to the web page or number of queries where the restaurant was retrieved.

Explicit attributes are the more relevant and accurate, since the user states his opinion. However, it heavily annoys the user and, therefore, it is not always possible to obtain [Carrol and Rosson, 1987]. Implicit attributes have a quite minor accuracy, but there is no annoyance of the user, since the system just tracks his behavior and learns about it. Thus, we will deal with both kinds of feedback and we will represent the case interests through both explicit and implicit attributes.

Finally, if a case represents a user interests, the complete case base is the user profile representation. Each personalized agent keeps, then, a case base that is the representation of the user on behalf the agent acts on.

3.2.3.2. The Case Base Adaptation

The main idea of the CBR is to solve new problems adapting the solution of old ones. It should be remarked that with a larger set of cases the system gives better results. However, several papers claim that when the case base reaches a number of cases, the performance of the system remains the same and sometimes decreases [Leake and Wilson, 1998].

One of the main drawbacks of the CBR is, then, the dimensionality problem: the uncontrolled growth of the case bases may result in the degradation of the performance of the system as a direct consequence of the increased cost in accessing memory.

Therefore, there is a need for a technique that controls the case base dimensionality forgetting the irrelevant cases. Some approaches handle this problem by storing new cases selectively (for example only when the existing cases in memory lead to a classification error) and deleting cases occasionally [Kibler and Aha, 1988]. Other approaches incorporate a restricted expressiveness policy into the indexing scheme by planing an upper bound on the size of a case that can be matched [Francis and Ram, 1993].

In the context of personalized agents, several adaptation techniques have been developed to handle the change of the human interests over time. We propose the drift attribute. The drift attribute is our proposal to adapt the user interests over time and to solve the dimensionality problem of the CBR systems. The drift attribute is one of the interest attributes (see Figure 16) and its function is aging the cases of the case base like weighting the interests in a gradual forgetting function (see section 2.8.7).

The drift attribute approach works as follows:

- The drift attribute is a value between 0 and 1.

- New items are inserted in the case base with the maximum drift attribute. The items are incorporated to the case base when the user shows some interest about them. Thus, the drift attribute is set initially to be the maximum.

- The value of the drift attribute is decreased over time emulating the gradual process where people forget interests. The decreasing function is a simple function where the drift attribute $\phi(q)$ of a case q is decreased multiplying the last drift value for a factor δ between 0 and 1 (see Equation 1). But, when is the decreasing function applied? We have to take into account that the frequency wherewith users interact with the system is very different. Therefore, the decreasing function should depend on the user interaction instead of days or weeks. For example, the system could decrease the drift attributes when a new item is incorporated to the case base or every time the user log into the system.

$$\phi(q) = \phi(q) * \delta$$

Equation 1. Decreasing function of the drift attribute

- The value of the drift attribute is increased (reward) if the retrieved case results in a successful recommendation. The rewarding function is as simple as the decreasing one. The drift attribute $\phi(q)$ of a case q is increased dividing the last drift value for a factor λ between 0 and 1 (see Equation 2).

$$\phi(q) = \phi(q) / \lambda$$

Equation 2. Rewarding function of the drift attribute

When a case achieves a drift value under a threshold, it is discarded. If the drift value is low enough, it does not make sense to retain the item in the case base. The confidence value of interest that this item gives is insignificant and it is a useless case that only contributes to increase the dimensionality of the case base and to decrease the performance of the system. Therefore, removing cases with a low drift value is the best solution to automatically control the dimensionality of the case base.

Initiating the case base needs a setup phase where the parameters are tuned to achieve the best performance of the system. In other words, we will obtain different results changing the rewarding function, the decreasing function and the threshold. Finding out the optimal values is an empirical task based on metrics to evaluate the system.

Some researchers apply the gradual forgetting function to their systems to adapt the user profile to the new interests. However, they have a weight/age for all the items of a given topic and it is modified when some event affects one of the items in the topic. We think that such functions reduce the performance of the system, especially when the same topic gathers a large set of items. Because if in the same interest topic there is one single item that the user is interested in, the topic never drifts, even if the user is not interested in all the other items in the set. Alternatively, in our approach we assign a weight to each item, thus, every case is treated individually and we solve this problem.

3.2.3.3. Initial Profile Generation

It is desirable to know as much as possible from the user so that the agents provide satisfactory results from the very beginning. In our approach, the training set seems to be the best technique to generate the initial profile, since the training set list of items given by the user with the appropriate attributes of interest can be the initial case base.

Analyzing the initial profile generation techniques proposed in the state of the art, we could see different advantages and drawbacks. In the manual generation, the user tailor his profile, thus it is a really transparent method. But the user is annoyed and it is difficult for the user to define its preferences explicitly. The empty approach has a potential long time to know the user preferences, that is, the initial recommendations have a low quality. But in this case, the user is not annoyed. The stereotyping approach interviews the user with a quick manual questionnaire only with a little annoyance, but people is reluctant to give personal data. Typically, the user does not fill the questionnaire or provides false data. The training set approach totally depends on the profile learning technique, since the user just gives a list of items that he likes and/or dislikes, and the learning technique generates the profile. There is a little annoyance of the user and the user easily defines its preferences.

3.2.4. The Retrieve Phase

In CBR terminology, the retrieval task starts with a new problem description and ends when a set of best matching previous cases has been found. This task can be discomposed in two parts: first, calculating the similarity between the new case and the cases in the case base; and second, selecting the best group of similar cases.

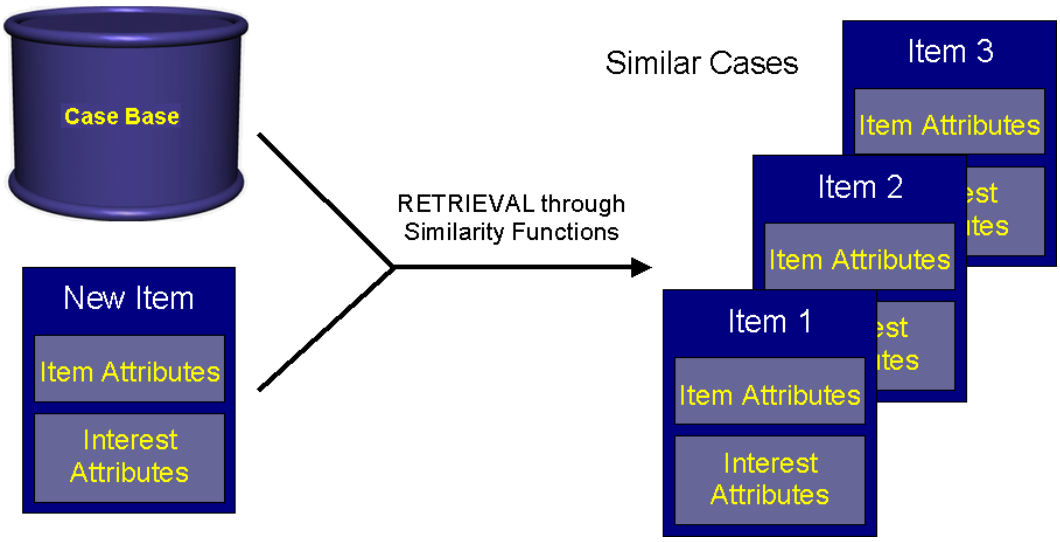

Figure 17. Retrieve Phase

When we apply CBR to personalization, this phase has the same purpose, but instead of retrieving similar problems, the system retrieves similar items. Thus, the retrieval task ends when a set of best matching previous items has been found (see Figure 17).

3.2.4.1. Similarity Function

The most interesting concept of the CBR is similarity. The idea is to define how similar are two cases based on their item attributes. In this way, given a case, we can get an ordered list of similar cases. Taking advantage of this concept, when a user likes an item, we can recommend him a list of similar ones that the user should like.

One way to obtain the similarity value between two cases is to predefine a numerical judgement called the similarity metric. A similarity metric can be any function that takes two entities and returns a value reflecting their similarity with respect to a given goal. Typically, researchers define the similarity between two cases as a combination of the similarities between the different attributes of the cases, since we cannot make use of all the case features available in a database in a uniform way. The system does not have knowledge about the meaning of each attribute, thus, it is usual to predefine how to handle the different features.

Moreover, in most cases overall similarity of features was a poor metric for providing examples, because users attached different significance to features depending on their goals. For example, if your goal is to buy a car that will pull a big trailer, you will weight engine size more heavily when comparing cars than other feature such as fuel consume. So, the system should regard engine size as more significant in assessing similarity in this context.

Thus, to compare two cases based on their attributes, the global similarity function should be a weighted ponderation of the different attribute similarities based on the user interests. The similarity between cases *q* and *c* is:

$$\sigma(q,c) = \sum_{i=1}^{n} w_i \sigma_i(q_i, c_i)$$

Equation 3. Global Similarity Function between cases *q* and *c*

where $q_1,\ldots,q_n$ are the item attributes of case q, $c_1,\ldots,c_n$ are the item attributes of case c, σ_i are the different attribute similarities and w_i are the weights of the ponderation. Typically, the weights must sum 1 and a human expert should provide them.

Thus, the key of the global similarity function is to define the different attribute similarities (σ_i). Certainly, σ_i depends on the type of attribute. We can classify the attributes in numerical or labeled. Numerical attributes contain a value and labeled attributes contain a text. For example, the age of a person is a numerical attribute because contain a number of years, on the other hand, the trademark of a product is a labeled attribute because contain a name of a company.

In our case base we deal with both numerical and labeled attributes, thus, we need to consider similarity metrics for both kind of attributes.

Regarding numerical attributes, most of the similarity metrics are based on the two following approaches:

- The most commonly similarity metric used in CBR systems is the difference between the values of the two attributes (q_i and c_i) normalized over [0,1] and relativized to the range of the possible attribute values [max_i, min_i] (see Equation 4). This is a suitable metric for attributes that have possible values ordered with a proportional difference, that is, values that follows a linear function.

$$\sigma(q_i, c_i) = 1 - \frac{q_i - c_i}{max_i - min_i}$$

Equation 4. Relative linear similarity function between two numerical attributes

- But sometimes the difference between two attribute values does not follow a linear function. For instance, to compare the similarity between two attributes representing the price of different products, we cannot consider the same difference between 1 and 100 than between 10000 to 10100. Thus, we need a non-linear similarity function (see Figure 18).

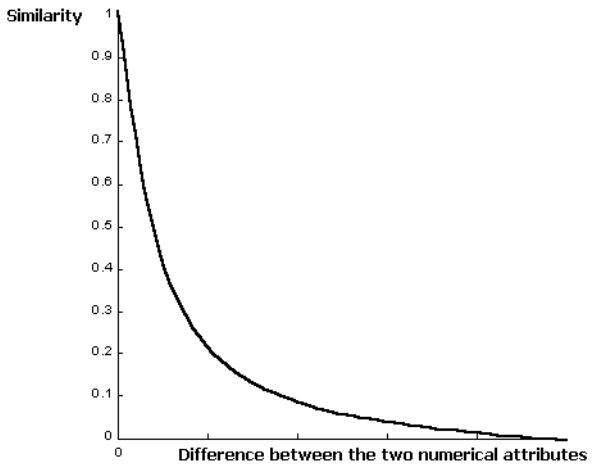

Figure 18. Non-linear similarity function between two numerical attributes

Regarding labeled attributes, defining a metric is a difficult task, since you cannot define a similarity function that compare text values. There are two main approaches to compare labeled attributes depending whether there is a given order between the attribute values or not. If the possible text values have an order (e.g., small, normal and big), you can assign a numerical value to each label and compare the text values with a numerical function (e.g., small=0, normal=5 and big=10, and the similarity function=1-(x-y/10)). Otherwise, when there is not an ordered relation between the possible text values, you must define a similarity table explicitly. The rows and columns of the table represent the attribute labels and the cells represent the similarity value between the two labels. The example shown in Figure 19 is a similarity table of the attribute work environment of a computer. The possible text values are house, industrial and spatial and the table gives the similarities between these possible labels. The restaurant recommender system Entree [Burke, 2000] uses such similarity tables. The main drawback of this approach is that it involves additional knowledge engineering: for n possible attribute values, a n2 adjacency table must be generated. This is particularly a problem in domains that the attributes have a large number of possible labels.

case 1 / case 2	HOME	INDUSTRIAL	SPATIAL
HOME	1	0,4	0
INDUSTRIAL	0,8	1	0,2
SPATIAL	0,6	0,8	1

Figure 19. Example of similarity table between two labeled attributes

3.2.4.2. Item Selection

Once the similarities between the new case and the cases in the case base are calculated, a set of best matches is chosen. There are two different methods to select the most similar cases:

- Selecting all the cases with a similarity over a threshold. The main drawback of this approach is defining the threshold. With a high selection threshold we just select the most similar cases, but there is the possibility to select only a few cases or none. With a low selection threshold we select many cases, maybe too many cases and cases with a very low similarity. Thus, tuning the threshold is a difficult task.

- Selecting the most similar n cases. The main drawback of this approach is defining the number *n* of cases to select. With a few cases you could ignore relevant cases and with a large number of cases you could select cases with a very low similarity.

However, the two approaches have a good performance and are used actually. In our approach, we want to deal with a hybrid of both approaches. We want to select the n best cases provided that they exceed a minimum selection threshold.

3.2.5. The Reuse Phase

The reuse phase consists in adapting the old solutions of the retrieved cases to the new problem based on the differences among them. Once the system has retrieved a set of previous items (the most similar ones), the system knows the interest of the user about similar items though the interest attributes of the case. Assuming that the interest of the user about a new item is similar to the interest of the user about similar items, in the reuse phase, the system calculates a interest

confidence value of the new item. This value is used to decide whether recommending the new item to the user.

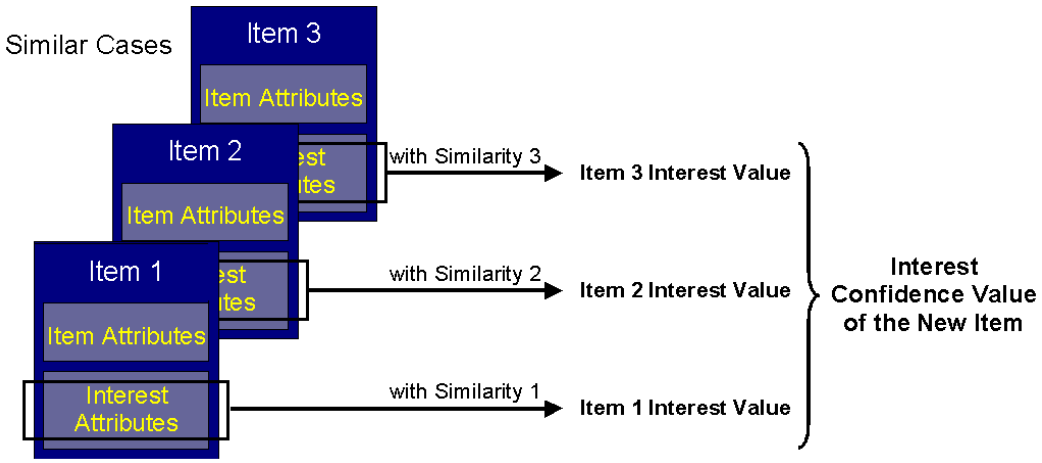

Figure 20. Reuse Phase

The interest confidence value is a composite of the item interest value of the similar items selected in the retrieve phase (see Figure 20). So, we calculate it in a two step process. First, the item interest value of each case is computed based on its interests attributes (see Equation 5). Given the following case representation:

Case / Item	Problem / Item Attributes	Solution / Interest Attributes		
		Explicit	Implicit	Drift Attribute
p	$p_1, p_2, ..., p_n$	$p^e_1, p^e_2, ..., p^e_{n_e}$	$p^i_1, p^i_2, ..., p^i_{n_i}$	da_p
q	$q_1, q_2, ..., q_n$	$q^e_1, q^e_2, ..., q^e_{n_e}$	$q^i_1, q^i_2, ..., q^i_{n_i}$	da_q

where p and q are items, $p_1, p_2, ..., p_n$ and $q_1, q_2, ..., q_n$ are the item attributes, $p^e_1, p^e_{2,...,} p^e_{ne}$ and $q^e_1, q^e_{2,...,} q^e_{ne}$ are explicit attributes, $p^i_1, p^i_{2,...,} p^i_{ni}$ and $q^i_1, q^i_{2,...,} q^i_{ni}$ are implicit attributes and da_p and da_q are drift attributes; the item interest value of the item p is calculated as follows:

$$V_p = da_p * g(f^e(p^e_1, ..., p^e_{n_e}), f^i(p^i_1, ..., p^i_{ni}))$$

Equation 5. Item Interest Value Function

where g is the function that combines the explicit and implicit attributes, f^e is the function that combines the explicit attributes and f^i is the function that combines the implicit attributes.

The explicit attributes are the most relevant, since the user stated its opinion. Therefore, g is mostly based on the explicit attributes. The implicit information is also used, but with a lower

contribution. The role of the drift attribute da_p is also very important to compute the item interest value, since it represents the present condition of the items.

Second, the interest confidence value *I* of a new item *r* is a weighted ponderation function of the item interest value of each similar item:

$$I_r = \overset{x}{F}_{i=1}(S_i * V_i)$$

Equation 6. Interest Confidence Value Function

where *x* is the number of similar items, *F* is the function that combines the different similar items, S_i is the similarity between the item r and the item i and V_i is the item interest value of the item *i*. In this way, the most similar items are the most relevant in the final result.

Finally, if the interest confidence value of the new item is greater than a certain value (a confidence threshold), the item is recommended to the user. Otherwise, the system ignores it, the CBR cycle finalizes and, therefore, there is no recommendation to the user. The item has no interest enough to the user and the agent should not disturb him/her with it.

3.2.6. The Revise Phase

The revise phase consists on evaluating the case solution generated by the reuse phase and learning about it. If the result is successful, then the system learns from the success (case retainment), otherwise it is necessary to repair the case solution using domain-specific knowledge. The revise phase is usually a step outside the CBR system, since it involves the application of the suggested solution to the real problem and, typically, the evaluation of an expert with knowledge about the domain.

When we apply CBR to personalization, in the revise phase, as from the relevance feedback of the user, the system is able to evaluate the interest of the user about the recommended item. The idea is tracking the user interaction filling the interest attributes of the item (case).

As shown in Figure 16, the interest attributes are distributed in two main groups: the implicit attributes and the explicit attributes. Obviously, the implicit attributes come up from the implicit feedback of the user, and the explicit attributes come up from the explicit feedback. The idea is to know the interest of the user based on a hybrid relevance feedback system. The user is explicitly inquired about the new item, but taking into account that users are very reluctant to give explicit feedback [Carroll and Rosson, 1987], the system tracks the user interaction with the system and try to conclude additional information.

In CBR systems the solution is successful or wrong. When the solution is successful, the system retains the case inserting it to the case base. But when the solution fails, the system is interested in retaining the reason of the failure and the good solution, thus, there is an investigation task to find out additional information about the case. In the personalization field, the interest of the user can be also positive or negative, but contrarily to this situation, the system is interested in retaining both the positive and the negative feedback. It is equally important to keep positive information about the interests of the user than negative information, since it is useful to know what the user does “love” and what the user does “hate”. Thus, in this approach, there is not an investigation task to know why the user is not interested in the new item, we just retain that the

item is not interesting for the user. Therefore, avoiding the investigation task, typically accomplished by a human expert, we achieve a completely automatic system.

3.2.7. The Retain Phase

In general CBR systems, the retain phase is the process of incorporating what is useful to retain from the new problem solving episode into the existing knowledge. It involves selecting which information from the case to retain, in what form to retain it, how to index the case for later retrieval from similar problems, and how to integrate the new case in the memory structure.

When we apply CBR to personalization, the task of the retain phase is almost the same. The new item is inserted to the case base with the interest attributes added in the revise phase. If the user did not give neither explicit feedback nor implicit feedback, and, therefore, the item has no interest attributes, the case is not introduced in the case base. Otherwise, items with positive interest and items with negative interest are retained.

Moreover, when the user gives explicit or implicit feedback about an existing item of the case base, the case is updated. For example, if the user consults the web page of an item, the interest attribute representing the number of visits to the web page and the attribute representing the time spent watching the web page are increased.

In order to control the case base dimensionality, it is also important to know whether the user never gives new feedback about items in the case base. In such case, it is necessary forget these interests with time. This problem is solved with the drift attribute explained section 3.2.3.2.

3.2.8. Related Work

A few groups of researchers investigate the application of CBR concepts and techniques to personalization. Cunningham et al. apply retrieval and adaptation techniques from CBR to the intelligent product recommendation agents, in particular to their WebSell system [Cunningham, 2000]. Like in our approach, the core of such applications is a product database that describes the specific features of each available product. When applying CBR, WebSell treats the product database as a case base, i.e., each product record in the database is interpreted as a case. During the case retrieval phase, product cases are also retrieved based on the similarity between the product features and the requirements elicited by the user. The similarity encodes the knowledge to assess whether a product is suitable for the customer's requirements. The ideas of such approach are more or less the same ideas that we want to apply. However, the Cunningham group is not concerned about the change of the user interests over time.

All "FindMe" systems [Burke et al., 1997] implement a similar CBR. They contain a database, they retrieve from it items that meet certain constraints, and they rank the retrieved results by some criteria. For instance, the restaurant recommender Entree [Burke, 2000] makes its recommendations by finding restaurants in a new city similar to restaurants the user knows and likes. The system allows users to navigate by stating their preferences with respect to a given restaurant, thereby refining their search criteria.

The drift attribute is a new idea that we introduce to the CBR applied to the personalization, thus, there is no work in this concept. But, this idea is based on the gradual forgetting function that some researchers apply to other CBR systems. As seen in section 2.8.7, this concept was introduced by [Webb and Kuzmycz, 1996] and later applied in systems such as SiteIF [Stefani and Strappavara, 1998] or LaboUr [Schwab et al., 2001].

3.2.9. Conclusions

In this section we have presented a new approach to personalization based on CBR. The first contribution of this approach is the case representation. Each case of the case base consists in a first set of attributes describing the item (as a problem definition) and a second set of attributes describing the interest of the user (as a problem solution). Supposing that the user has a similar opinion about similar items, we can use the interest attributes of the old cases to guess whether a new item is interesting for the user.

Another contribution of this approach is the drift attribute. The drift attribute represents the age of the case in the case base and lets to the personal agent distinguish between current and old interests. Consequently, the newest cases will give more confidence to the recommendations than the oldest ones. To control the velocity of adaptation to the new interests, we can tune the decreasing function, the rewarding function and/or the drift threshold. The most important advantage of applying the drift attribute is that the dimensionality problem is solved. Drift cases are deleted and the number of cases in the case base becomes stable while the performance of the system is maintained.

The main drawback of CBR applied to personalization is that the agent will tend to over-specialize, not finding new items outside the case base, since it is a content-based filtering methodology (see section 2.3.2). However, this drawback is solved with the integration of the personalized agent in its environment and using the notion of trust in the collaborative world.

3.3. Trust in the Collaborative World

In the real world, people ask to their friends for interesting items. For example, a common situation is when somebody wants to try a new restaurant and ask for advice to a friend. Another common situation is when you discover a new restaurant and you want to know the opinion of your friends about it or when somebody tell you something about a new restaurant, you want to verify this information with your friends. If they know it, they can give you their opinion, and if they do not know it, as from the features of the restaurant (e.g., cuisine, price,...) they can guess an opinion. But people do not ask for advice to any person. People only ask for advice to friends with similar tastes and interests, friends that we can trust in. And, how do people know whether other people have similar tastes and interests? Typically, through interaction. If you want to know the tastes and interests of another person, you ask him for his opinion. For example, in the restaurants domain, you ask him for his opinion about restaurants that you love or about restaurants that you hate. If he has a similar opinion, you consider him a person with similar preferences.

Trust in the collaborative world is a new approach based on the agent's theory. Such approach let us to apply all these concepts from the real world to the personal agents. Mainly, we provide personal agents with a technology to evaluate whether they can rely in the other agents to recommend to the user. Thus, a personal agent can look for similar agents that advice him.

The over-specialization problem of the CBR approach is solved. Taking advantage of the collaborative world, similar agents advice the recommendation of new items. Typically, many of these items are not similar to the items contained in the profile. Such situation is typical in the real world: similar friends recommend you items that you never have seen before.

The approach of trust in the collaborative world approach is presented as follows. Section 1.2.1 shows how personal agents take advantage of the collaborative world. Section 1.2.2 introduces the concept of trust and its implementation. With the capability of trust, a new information

filtering method comes up. It is explained in section 1.2.3. Finally, section 1.2.4 concludes the approach.

3.3.1. The Collaborative World

All the personal agents acting on the system compose the collaborative world. Each personal agent represents a user and its goal is to recommend to the user interesting items. The agent can just recommend items based on the previous evaluated items trying to find out similarities on the content. But, he can also take advantage of the other agents. The main idea of this approach consists in thinking about the other agents as personal entities in which you can rely on or not. We suppose that personal agents with similar opinions about the same products are initially similar. But, taking into account that two agents are never equal, it is quite possible that they give bad recommendations. Thus, if agents keep a trust value about other agents, they can modify it with time depending on their recommendations and, then, ignore its opinion. Therefore, with this approach, an agent cannot rely in another similar one.

Current collaborative filtering systems just recommend new items based on the similarity between profiles. This means that agents with similar profiles exchange recommendations. However, when a similar agent gives unsuccessful advice, there is no way to ignore it. Over and over again this agent causes a descent in the other agent performance. Applying the trust in the collaborative world approach, this problem is solved. When a similar agent gives frustrated recommendations, the other agent can decrease the trust in this agent and ignore its advice in the future.

3.3.2. The Trust

Taking into account the lack of altruism in a distributed artificial intelligence systems, Marsh proposes the concept of trust to make our agents less vulnerable to others [Marsh, 1994]. Trust is fundamental for any kind of action in an uncertain world; in particular it is crucial for any form of collaboration with other autonomous agents. There is no a definition for trust, but Gambetta provides a basic definition in his book that has been accepted by the great majority of the authors [Gambetta, 1990]:

"Trust is the subjective probability by which and individual A, expects that another individual B, performs a given action on which its welfare depends"

Castelfranchi and Falcone agree with this definition and emphasize that trust is basically an estimation, an opinion, an evaluation, i.e. a belief [Castelfranchi and Falcone, 1998]. Elofson gives another definition closer to our approach [Elofson, 1998]. He claims that observations are important for trust, and he defines trust as:

"Trust is the outcome of observations leading to the belief that the actions of another may be relied upon, without explicit guarantee, to achieve a goal in a risky situation"

Elofson notes that trust can be developed over time as the outcome of a series of confirming observations (also called the dynamics of trust). From his experimental work, Elofson concludes that information regarding the reasoning process of an agent, more than the actual conclusions of that agent affect the trust in the conclusions of that agent.

Each event that can influence the degree of trust is interpreted by the agent to be either a negative or a positive experience. If the event is interpreted to be a negative experience the agent will loose his trust to some degree and if it is interpreted to be positive, the agent gain

trust to some degree. The degree to which the trust is changed depends on the trust model used by the agent. This implies that the trusting agent performs a form of continual verification and validation of the subject of trust over time.

In our approach, trust represents the level of reliability in the other personal agent, thus, reliable agents are agents that represent similar users, and unreliable agents are agents that represent users with different interest. Moreover, the concept of trust can help us to ensure that our personal agent is more robust with respect to interactions with other agents. The agents only ask for advice to other agents with a trust value above a threshold. Defining the trust threshold is a difficult task. An agent with a high threshold maybe has no reliable agents and, with a low threshold, the agent will rely in not very similar agents. Defining the trust threshold is an empirical task and should be matter of study.

Each personal agent has its own personal contact list the agent keeps a the id of the other agents, how to contact them and the trust value. One of the key issues for the design of the personalized agent is how trust is represented within the agent, and how the effect of experiences updates it. Representations can be qualitative, using specific qualitative labels [Jonker and Treur, 1999], or quantitative, using numbers as representation. In our approach, we represent the trust as a confidence value, ranged from 0 to 1.

3.3.3. The Opinion-based Information Filtering Method

A new information filtering method comes up with trust in the collaborative world of autonomous agents. Taking advantage of the communication among them, a personal agent can ask for the opinion of a given item to the other agents. It differs from the typical collaborative filtering approach in the way that the agent does not ask for a recommendation, the agent ask for an opinion. The opinion is the interest that the other agent thinks that his user has about the given item. Instead of using this opinion directly as a recommendation, the agent includes it in the own reasoning and with other agents opinions in order to decide whether recommending a given item. We call this new filtering method the *opinion-based information filtering method*.

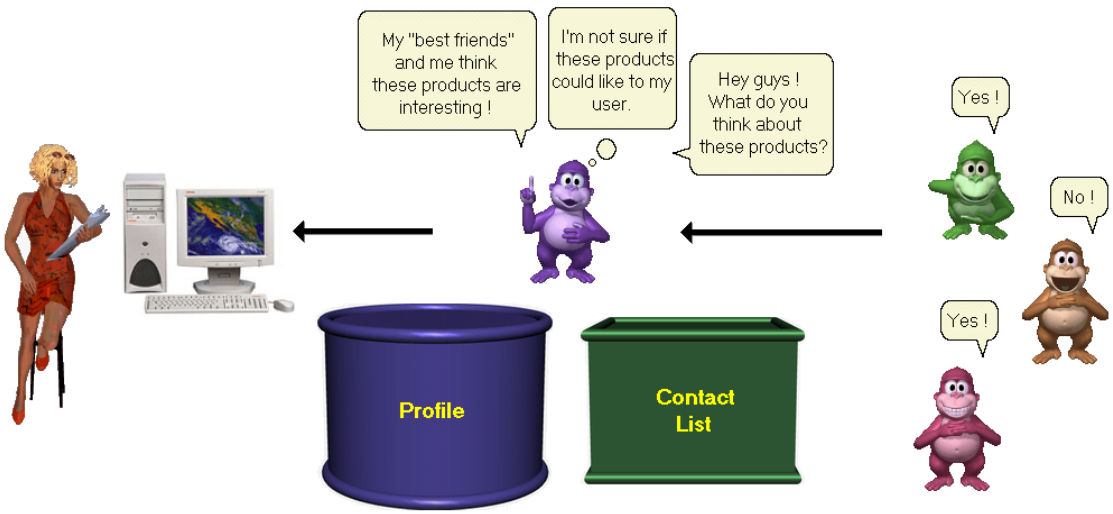

Figure 21. Information Filtering based on Opinion

With this new information filtering method several questions come up to our minds. Firstly, when the personal agent asks for the opinion? About which items?
When the agent is not sure about a recommendation or he just discovers a new item, he asks for the opinion to the reliable agents and uses their confidence values to conclude whether the item is interesting for the user (see Figure 21).

Who are the interrogated agents?
Taking into account the main feature of the trust in the collaborative world approach, this question has an easy answer. Agents just ask for the opinion to the reliable friends, that is, the agents in the contact list with a higher trust value. Once the personal agent has the opinion of the other agents, a consensus has to be achieved. Several consensus measures like [de la Rosa, 1994] will be analyzed to accomplish this task.

How do agents create and maintain a trust value of the other agents?
Agents are compared proactively through the interaction ("playing agents" [Steels and Vogt, 1997]). For instance, asking for the opinion about items that are already known by the user. The agent asks for the opinion about the items that the user "loves" (a high confidence of interest) or "hates" (a confidence of interest near to zero) to the enquired agent. The agent gathers the opinions and infers a trust value. This last step is a difficult task. Aggregation techniques like [Torra, 2001] will be analyzed for this purpose.

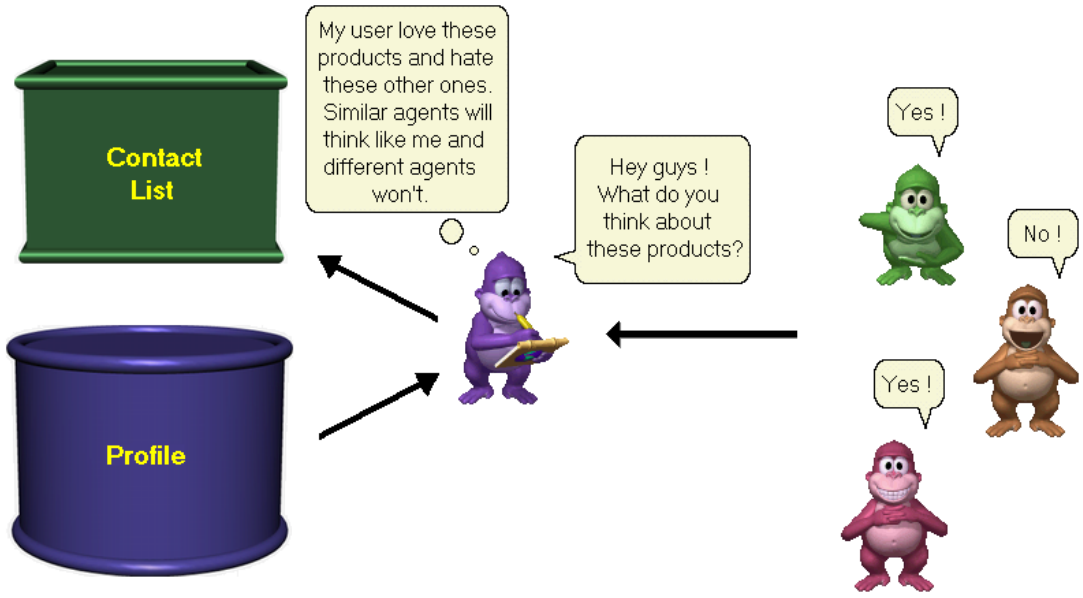

Figure 22. "Playing Agents"

Applying an agent-based approach with the trust in the collaborative world to personalization, the typical information filtering methods (content-based and collaborative filtering) can be also applied. The content-based filtering method has the same performance with this approach, but the collaborative filtering method is improved, since personal agents just believes in the recommendations of the agents with a high trusting value. Finally, we get a hybrid approach among opinion-based, content-based and collaborative filtering.

3.3.4. Related Work

Trust in the collaborative world have been applied to other fields like multi-agent systems [Zacharia, 1999], [Schillo et al., 2000], [Yu and Singh, 2000] and [Wu and Sun, 2001] with really good results. Some new investigations are applying trust on market places [Sabater and Sierra, 2000] and other problem solving MAS organizations [Ontañon and Plaza, 2001]. But the approach presented here applied to personalized agents is new. We want to exhibit that applying these concepts the performance of the personalized agent is improved by proactivity.

3.3.5. Conclusions

Trust in the collaborative world is a new approach that seems to be a suitable for personalized agents. Like in the real world, agents rely in some agents and mistrust in other ones to achieve a purpose. If we provide personal agents with a technology to evaluate and trust in the other agents, personal agents can exploit the collaborative world with a better performance.

When we apply trust in the collaborative world to personal agents, a new information filtering method comes up. A personal agent can ask for the opinion of a given item to his reliable friends, and achieve a consensus to recommend to the user. The opinion-based filtering method can be considered as an evolution of the collaborative filtering methods due to the agent's world. But, if we consider that the hybrid approaches between content-based and collaborative filtering exhibit better results (see section 2.3.4), we can see this approach as an evolution of the information filtering methods in general (see Figure 23).

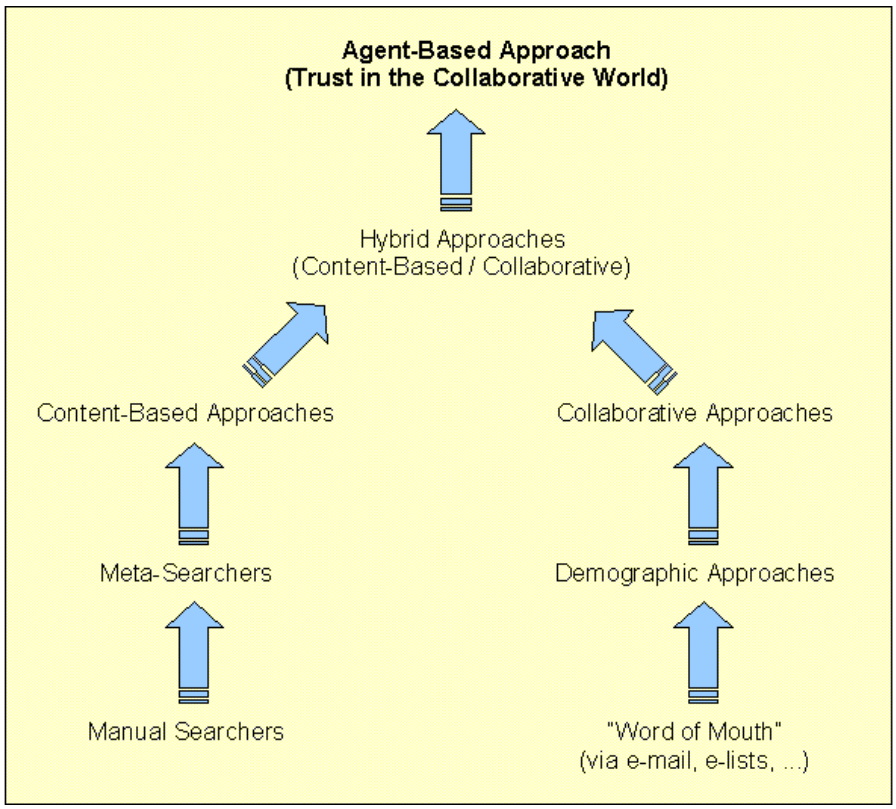

Figure 23. Information Filtering Evolution based on the Personalization

3.4. Conclusions

A new approach to personalized agents has been proposed based on CBR and trust in multi-agent systems. On one hand, CBR allows the utilization of knowledge of previously evaluated items for the user to guess the interest of the user about new items. The CBR nature provides user acceptance and easy maintenance to the system. The user profile is a case base. Each case is represented with a set of qualitative attributes describing the item and with a set of interest attributes describing the opinion of the user about them. In our approach, the agent asks for explicit opinions about the items and tracks the user navigation through the system in order to capture implicit opinion. Both, explicit and implicit feedback, constitute the interest attributes of the case. The initial case base is generated as from a training set of items that the user evaluates the first time he logs into the system. The profile needs an adaptation to the changing interests of the user. Our personal agent includes a drift attribute in the cases representing the age of the interest and that is used to control the case base dimensionality, one of the main drawbacks of other CBR systems.

On the other hand, trust in multi-agent systems is a technology that allows agents thinking about the others as personal entities in which you can rely on or not. The proposal of trust in the collaborative world makes personal agents more robust in front of the other agents. Trust in the collaborative world is proposed as the basis for a new information filtering method: the opinion-based filtering. The information filtering method based on the opinion is combined in a hybrid system with a content-based and collaborative filtering methods. Therefore, the sources of the recommendations to the user are:

- Content-based recommendations through the CBR approach with a very high confidence.
- Opinion-based recommendations through the CBR approach with not a very high confidence but highly recommended from your reliable friends.
- Collaborative recommendations through your reliable friends with a very high confidence.

To implement trust, the personal agent is integrated in a community of agents where a server agent provides information about items and contact information about other agents representing users. Each agent of the community is:

- Autonomous: the agent does not need the direct intervention of the user to operate and it controls its actions and internal state.
- Reactive: the agent percepts the environment and reacts when it changes. For example, when new agents are incorporated to the system the agent starts a conversation with them to know something about their interests, or when the server offers new items the agent analyzes them to know whether they are interesting for the user.
- Pro-active: the agent proactively interacts with other personal agents looking for similar users and it also interacts with the server agent looking for new interesting items.
- Has social abilities: the agent is capable to interact with other agents (other personal agents or the server agent) and the user.

The proposed approach is constrained to the domain where the user needs the items with a relative high frequency. For instance, buying products in the supermarket, renting videos or selecting restaurants, where frequent recommendations of these items are useful for the user. A more difficult challenge is presented for an item that the user needs less frequently and one at a time. For example, an automobile, a home loan or any other infrequently purchased item, where

the system will not be able to use market-basket or purchase history to make recommendations. Moreover, we see this approach as applicable to any domain with a large set of items, since a recommendation system is more useful when the domain is sufficiently complex and users would probably be unable to fully articulate their retrieval criteria. We here decided to implement our personalization approach to the restaurants domain, which fulfills such requirements.

It is actually hard to determine how well a personalized system works, since it involves purely subjective assessments. This contributes to the reluctance of the user to accept personalized systems. There is no technique that is the best, since there is no an objective comparative of the 37 analyzed systems in the state of the art. Each system evaluates different techniques in its framework and shows important results. However, it is impossible to generalize these results to all the frameworks. Most of the systems are analyzed with the logs technique (see section 2.11.1.3), since only important systems like Amazon or CDNow can show real and believable results due to its benefits.

To evaluate our system, we agree with the option introduced in NewT [Sheth and Maes, 1993] and later implement in other systems like CASMIR [Berney and Ferneley, 1999] and [Holte and Yan, 1996] in the use of a user simulator. A user simulator let us investigating with large collections of simulated users whose design was tailored to explore the desired issues. The main advantage of this approach is that the system enables to carry out large-scale experiments with guaranties of control, quickness and repeatability. The user simulator is really useful taking into account that to implement the CBR approach many parameters have to be tuned (i.e., similarity functions, drift attribute parameters,...).

In further work, the idea is transforming the personal agent to a personal ecosystem of agents where each agent just keeps information about one topic (see section 6.2). Thus, the ecosystem of agents will be composed by a large set of the agents, each one specialized on a topic, which interact among them and interact also with the agents of other users.

4. PRELIMINARY WORK

We have developed a preliminary work to exhibit the viability of Case-Based Reasoning (CBR) applied to recommendation systems. In particular, we implemented a restaurants recommender system on the Internet called GenialChef[1].

4.1. GenialChef

GenialChef is a preliminary work carried out in the e-commerce domain to help travelers to find their favorite restaurants. The main goal of GenialChef is provide users with a tool to easily find restaurants that satisfy their requirements and tastes.

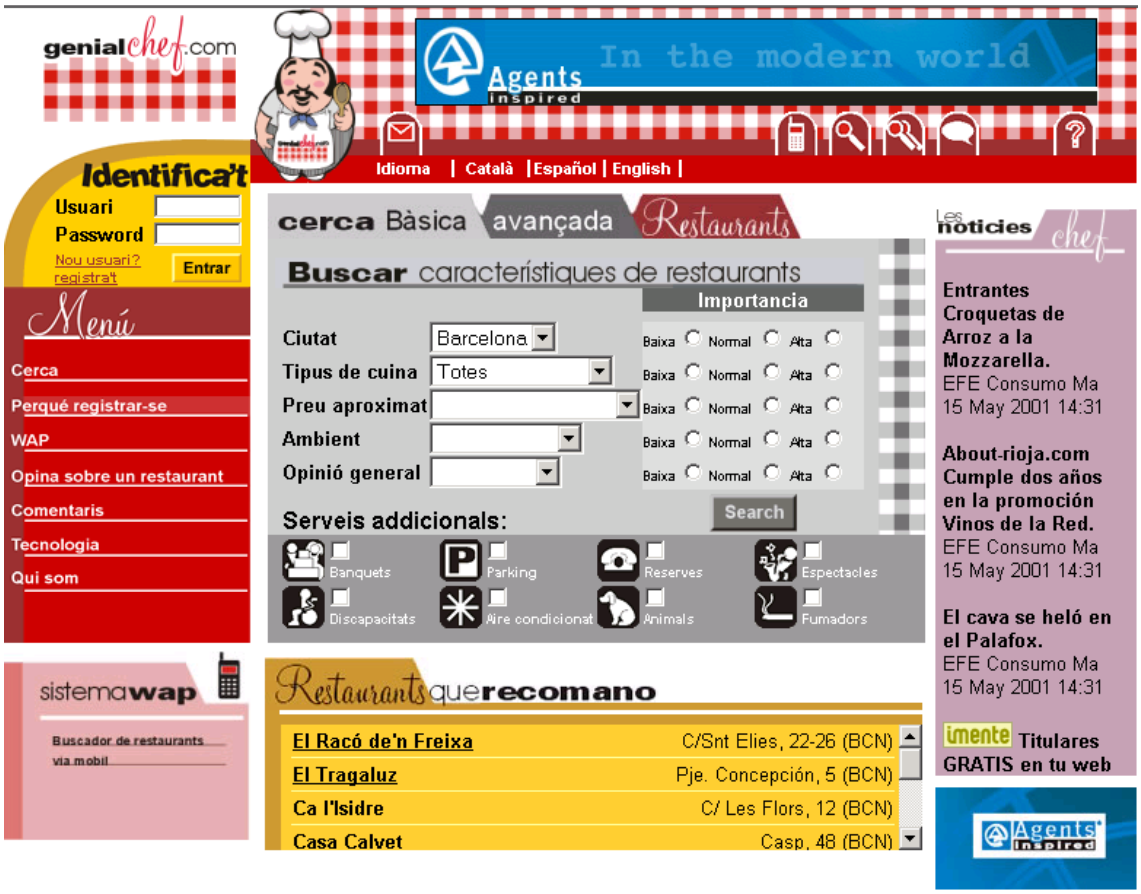

Figure 24. GenialChef Main Page

The great difference between GenialChef and other information gateways is the adaptability of the information showed in web pages: GenialChef personalizes the content based on user profiles and CBR technology.

[1] Soon available at http://www.genialchef.com

To achieve personalization through CBR, the restaurants and user profiles representation is a key task. Restaurants information is stored in a case base structured in representative attributes as cuisine, approximated price and address. User profiles are represented with personal case bases containing information about previously evaluated restaurants (see Figure 26) and past queries (see Figure 25).

Based on this represented information, GenialChef personalize the content of web pages through several functionalities:

- When the user logs into the system through the main page of GenialChef (see Figure 14), the system recommend interesting restaurants to the user based on his/her profile.

- When the user uses the searcher to look for interesting restaurants (see Figure 25), CBR is used to select the most similar restaurants based on similarity functions. Then, results are filtered as from the user profile.

- When the user uses the searcher to look for interesting restaurants, the searcher form is automatically filled based on past queries to accelerate the user search.

But GenialChef works also without a user registration. In this case, GenialChef is a typical gateway where users look for restaurants information. The only special feature is that the searcher uses CBR technology.

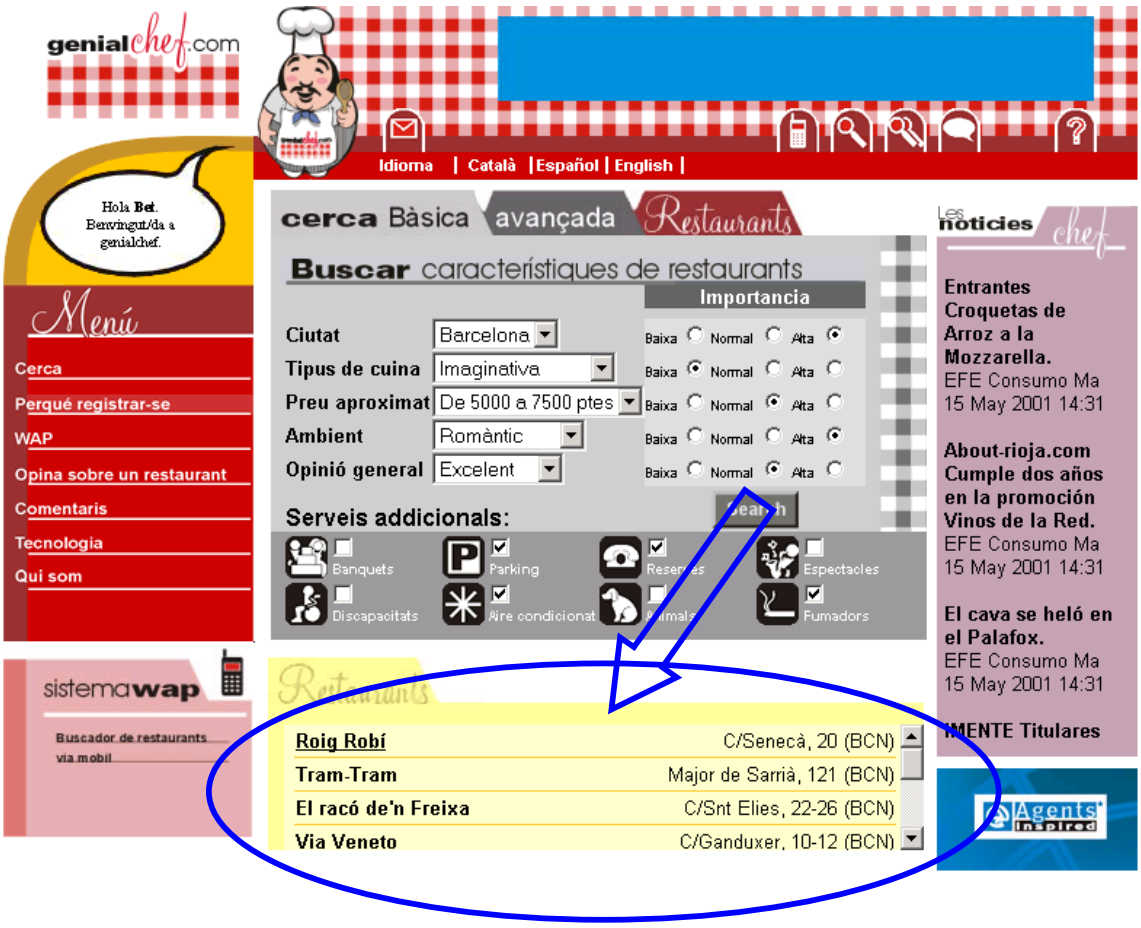

Figure 25. GenialChef Search

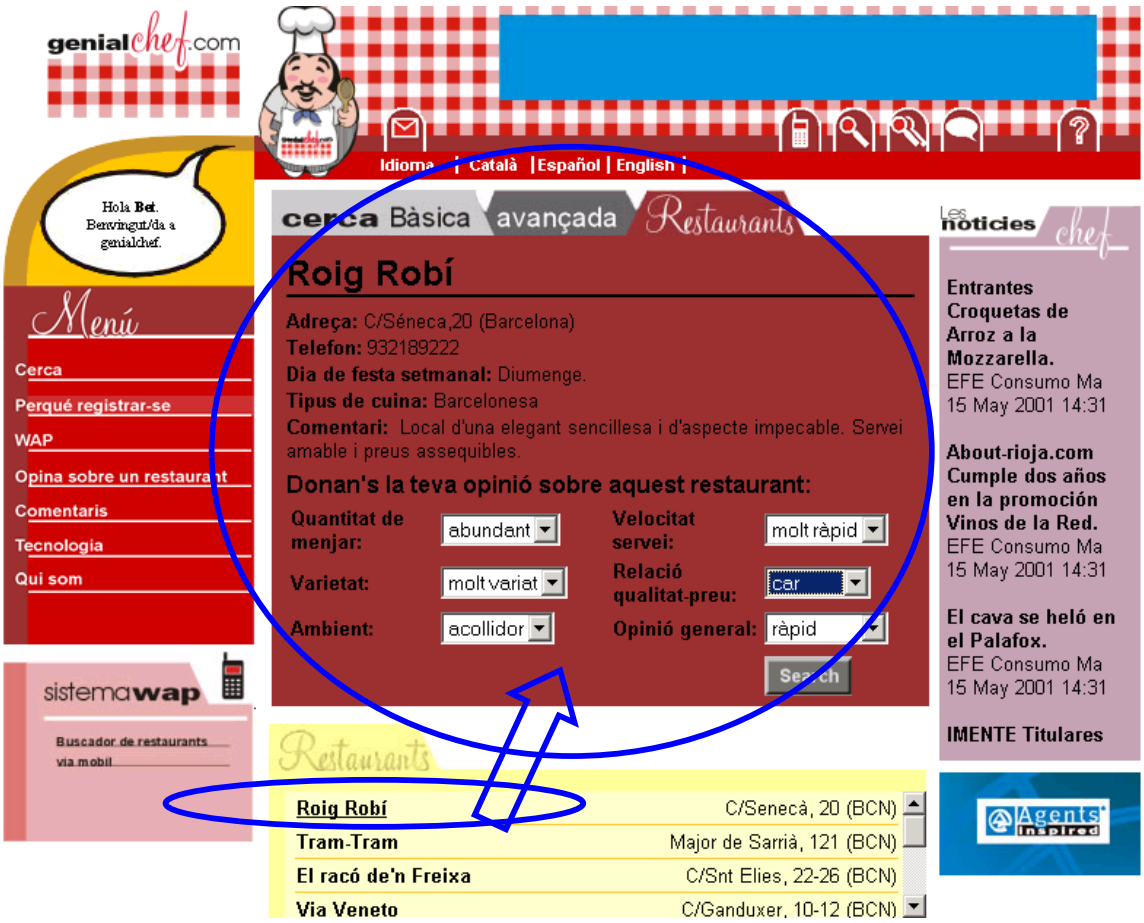

Figure 26. GenialChef Opinion Capture

For more information, GenialChef design and implementation is detailed in [Montaner et al., 2001].

4.2. Conclusions

The main features of GenialChef that we want to emphasize are:

- Anonymous and registered access.
- Users are treated individually: web pages are adapted with personal content based on the user profile.
- Taking advantage of CBR, restaurants are stored in a case base and the searcher queries it based on similarity functions.

- The user profiles are composed by previously evaluated restaurants and past queries. GenialChef captures the user personal opinions about visited restaurants and the user queries to find out what the user is interested in.

- As from the past queries stored in the user profile, the searcher form is automatically filled.

- Recommendations about interesting restaurants are proactively showed to the user according to the user tastes.

- System accessibility: users can access to GenialChef via web or via wireless devices like cellular phones or PDAs.

Therefore, when users visit another city and looks for a restaurant, they could access to GenialChef through the cellular phone and search a restaurant according to their interests.

GenialChef is just a start point to achieve the personalized system proposed in this document.

5. PLANNING

From the presentation of the thesis proposal (October 2001) until the presentation of the thesis (October 2003) we plan to execute 5 phases in this order: the agent platform, the CBR and trust approaches, the user simulator, experiments and results, and the thesis writing. In the following sections we briefly present the tasks that we need to carry out in each phase.

5.1. 1st Phase: The Platform

The first phase consists in developing the infrastructure of the system. Above all things, the technical issues have to be decided. The idea is programming the system with free software: a free database for the server agent like PostgreSQL[2] and a multi-platform programming language like Java[3], especially due to the distributed architecture of the system.

Here we have to decide whether we implement a new platform or we implement a MAS as from a free development tool. For the second option, tools like JADE[4] (Java Agent Development Framework), JAFMAS[5] (Java-based Agent Framework for Multiagent Systems) or Zeus[6] could be a good solution. If we implement a new system, the process can be divided in three parts: the server agent, the personal agent and the communications protocol. The item database, methods to query it and a set of methods to interact with personal agents compose the server agent. The personal agent is composed by the user interface and methods to interact with the server agent and other personal agents. Finally, a protocol to communicate personal agents with the server agent and with other personal agents has to be implemented. Of course, we will develop the system in compliance with the FIPA specifications[7].

The first phase will be carried out during 4 months, that is, from October 2001 until April 2002.

5.2. 2nd Phase: The CBR and Trust Approaches

Once we have the infrastructure of the system, where agents have an entity, we are going to implement the main part of the system: the CBR and the trust is the collaborative world approaches extensively exposed in this document.

The first step to develop the CBR approach is generating the case base. Taking into account that we want to construct a restaurants recommender, the best attributes to describe them have to be selected. Moreover, we have to define the algorithm and parameters to control the drift attribute. Then, we have to develop a module to accomplish the CBR cycle explained in section 3.2. The module consists in four parts: the retrieve engine, the reuse algorithm, the revise system and the retain step.

[2] http://www.ca.postgresql.org/
[3] http://java.sun.com/
[4] http://sharon.cselt.it/projects/jade/
[5] http://www.ececs.uc.edu/~abaker/JAFMAS/
[6] http://www.labs.bt.com/projects/agents.htm
[7] http://www.fipa.org/specifications/index.html

The implementation of the trust in the collaborative world starts providing the server agent with the agents list, and the personal agents with the contact list. Then, several algorithms should be developed to the new information filtering method: the selection of interrogated agents, the consensus among the opinions of the other agents, the proactive creation and maintenance of the contact list through aggregation methods among others.

We plan to carry out the CBR and the trust approaches in parallel during 8 months, that is, from April 2002 until October 2002.

5.3. 3rd Phase: The User Simulator

Once the system has been implemented, we need to test it and obtain results. Testing the system in a real environment is a really slow task. The idea is to implement a program that simulates large heterogeneous populations of users with different behaviors. The first part of the phase is to study the user simulators implemented in the state of the art (i.e., [Sheth and Maes, 1993], [Holte and Yan, 1996] and [Berney and Ferneley, 1999]). Then, based on the state of the art, a new simulator that satisfies our requirements have to be developed.

Planning the time to implement the user simulator is a difficult task, since it is the least defined phase. However, we expect to accomplish it in 4 months, that is, from October 2002 until February 2003.

5.4. 4th Phase: Experiments and Results

The experiments and its results on the new system are the most important part of the thesis. Using the user simulator, we should carry out large-scale experiments quickly. The results will allow us to tune the parameters of the system to improve the performance. Repeating the same experiments we can compare different tunings, different algorithms and even different personal agents. Therefore, we can conclude how many the performance of the system is increased with the new approach presented in this document.

We plan to carry out the experiments and analyze the results during 4 months, that is, from February 2003 until June 2003.

5.5. 5th Phase: Thesis Writing

This part encompasses the creation of the thesis document, the revision and the preparation of the thesis presentation. It is important to report the different phases during its implementation in order to facilitate the thesis writing. In this sense, we will consider the publication of results on AI congresses and journals specialized on the topic of the thesis. Interesting congresses could be Autonomous Agents and Multi-Agent Systems (AAMAS), Autonomous Agents, and International Joint Conference on Artificial Intelligence (IJCAI). Interesting journals could be AI Magazine, Autonomous Agents and Multi-Agent Systems, and Journal of Artificial Intelligence Research (JAIR). Thus, the creation of the final document will be only a report gathering.

The last phase needs 4 months, that is, from June 2003 until the thesis presentation.

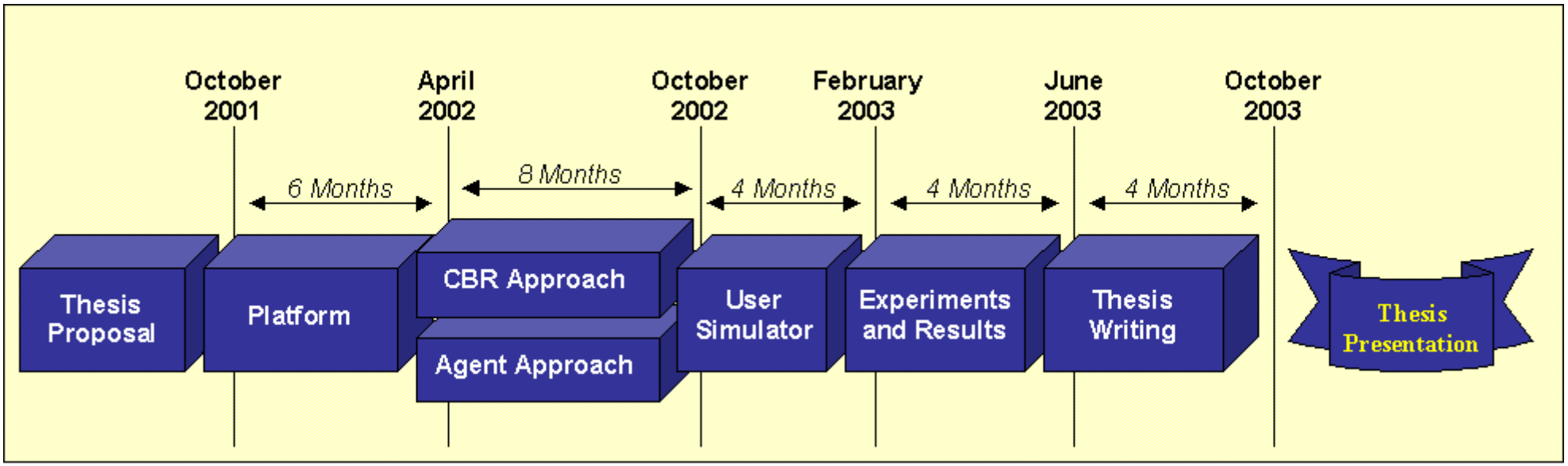

Figure 27. Thesis Planification

6. EXPECTED RESULTS AND FURTHER WORK

In this section, we briefly explain the results that we expect of this thesis and, to finish this proposal, we introduce the further work of this thesis.

6.1. Expected Results

The development of the work presented in this proposal will result on new contributions to the agents technology, but also to the machine learning community.

Regarding agents, we will stress the properties of:

- Privacy: the private user's data is encapsulated and protected by the agent.
- Peer to peer: its is a clear one to one concept, closer to having a human personal assistant that knows the user's tastes.
- Scalability: the more subjects and the more people, the more agents.
- Trust: to make our agents less vulnerable to others.
- Specialization and diversity: every agent knows about a specific subject and person, thus there are many points of view about the same potential recommendations.

Regarding machine learning community, the proposed CBR technology is a new advance towards the control of the dimensionality problem. Other important advances to recommender systems because the new CBR approach are:

- Maintainability: CBR systems do incremental learning, since the knowledge is easily increased just adding new cases in the case base.
- User acceptance: this is a key issue deploying AI technology: no system is useful unless its users accept its results. Users accept reasoning that seems natural to them.
- Temporal adaptation: drift interests are forgotten.

Finally, the application of the personalized agents to the restaurants domain will provide new improvements in the e-commerce field:

- Improves the exploitation of the one to one recommendation.
- Improves the merchandising.
- Improves the marketing reliability: improved market segmentation.

6.2. Further Work

This thesis is part of a broader project of personalization. We want to implement a personalized system where personal agents are experts in many topics. In this proposal, personal agents only handle one topic (i.e., restaurants). To achieve a multi-topic personal agent we want to create an ecosystem of agents, where each agent just handles one topic. Thus, instead of a personal agent we will have a personal ecosystem of agents.

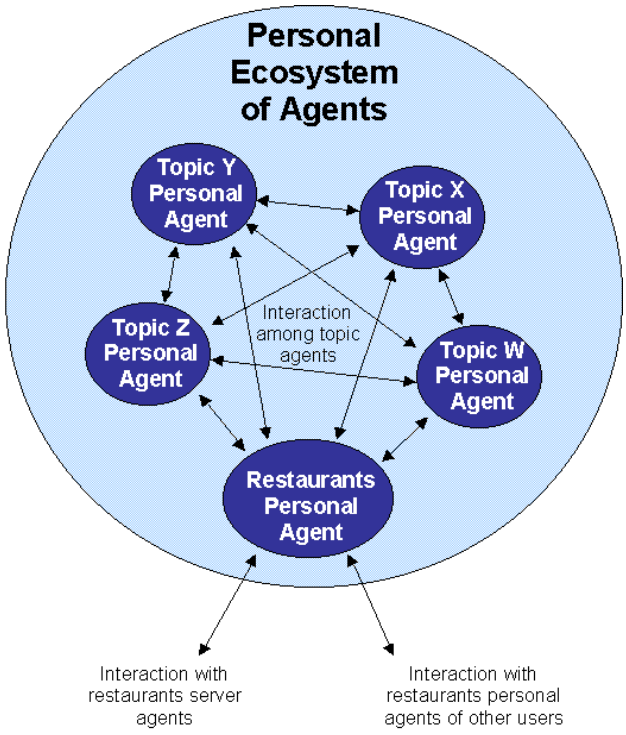

Figure 28. Further Work: a Personal Ecosystem of Agents

The ecosystem is composed by a large set of the personal agents introduced in this document, agents that only handle one topic. The topic personal agents, apart from interact with other personal agents of the same topic and with the server agent, interact with personal agents of the same user but of a different topic. Information about items of a topic could hardly constrain the user behavior in front of items of other topics. The collaboration of agents with different but closer topics can result in a more complete knowledge of the user interests. For example, a supermarket personal agent should inform to the restaurants personal agent about the preferences of the user in meal. If the user use to buy seafood in the supermarket, the restaurants personal agent could recommend seafood restaurants.

The main advantage of the personal ecosystem of agents is the open architecture. Personal agents with new topics can be easily added to the ecosystem.

The integration of all the topic personal agents and how to take advantage of their internal interaction are key issues of the further work.

7. REFERENCES

[Aamodt and Plaza, 1994]

Aamodt, A., Plaza, E., "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches" .AICom - Artificial Intelligence Communications, IOS Press, Vol. 7: 1, pp. 39-59. 1994.

[Amazon]

Amazon.com – http://www.amazon.com

[Ardissono et al., 1999]

Ardissono, L., Console, L., Torre, I.: "On the application of personalization techniques to news servers on the WWW". In: Lecture Notes in Artificial Intelligence N. 1792. Berlin: Springer Verlag, pp. 261--272. 1999.

[Armstrong et al., 1995]

Armstrong, R., Freitag, D., Joachims, T., Mitchell, T.: "Webwatcher: A learning apprentice for the world wide web". In 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous Distributed Environments, March 1995.

[Asnicar and Tasso, 1997]

Asnicar, F., Tasso, C. "ifWeb: a Prototype of user models based intelligent agent for document filtering and navigation in the world wide web", Sixth International Conference on User Modeling, Chia Laguna, Sardinia, Italy, 2-5 June 1997.

[Balabanovic and Shoham, 1995]

Balabanovic, M., Shoham, Y., "Learning Information retrieval Agents: Experiments with Automated Web Browsing", AAAI 1995 Spring Sumposium on Informatioin Gathering from Heterogeneous, Distributed Environments, Stanford, March 1995.

[Balabanovic and Shoham, 1997]

Balabanovic, M., Shoham, Y., "Combining Content-Based and Collaborative Recommendation". Communications of the ACM, March, 1997

[Basu et al., 1998]

Basu, C., Hirsh, H., Cohen, W.: "Recommendation as Classification: Using Social and Content-Based Information in Recommendation", Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI'98), AAAI Press/MIT Press, pp. 714-720, 1998.

[Berney and Ferneley, 1999]

Berney, B., Ferneley, E., "CASMIR: Information Retrieval Based on Collaborative User Profiling", In Proceedings of the Forth International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'99), The Practical Application Company Ltd, Lancashire, pp. 41-56. 1999.

[Billsus and Pazzani, 1999]

Billsus D., Pazzani M. J., "A Hybrid User Model for News Classification". In Kay J. (ed.),

UM99 User Modeling - Proceedings of the Seventh International Conference, pp. 99-108. Springer-Verlag, Wien, New York, 1999.

[Billsus and Pazzani, 1998]

Billsus, D., Pazzani, M., "Learning Collaborative Information Filters". Proceedings of the International Conference on Machine Learning. Morgan Kaufmann Publishers. Madison, Wisc, 1998.

[Boone, 1998]

Boone, G. "Concept Features in Re:Agent, an Intelligent Email Agent". The Second International Conference on Autonomous Agents (Agents '98), Minneapolis/St. Paul, 10-13, May 1998.

[Breese et al, 1998]

Breese, J., Heckerman, D., Kadie, C.. "Empirical analysis of predictive algorithms for collaborative filtering". In G. F. Cooper and S. Moral, editors, Uncertainty in Artificial Intelligence: Proceedings of the Fourteenth Conference, page 43--52. Morgan Kaufmann, San Francisco, July 1998.

[Buckley and Salton, 1995]

Buckley, C., Salton, G., "Optimization of relevance feedback weights". In Fox, Ed, Ingwersen, Peter, and Fidel, Raya (Editors), Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 351--357. 1995.

[Buckley et al., 1996]

Buckley, C., Singhal, A., Mitra, M., Salton, G., "New Retrieval Approaches Using SMART: TREC 4", in publishing of the Fourth Text Retrieval conference (TREC-4), NIST Special Publication, 1996.

[Burke et al., 1997]

Burke, R., Hammond, K., Young, B., "The FindMe Approach to Assisted Browsing". IEEE Expert, 12(4):32-40. May, 1997.

[Burke, 2000]

Burke, R., "Semantic ratings and heuristic similarity for collaborative filtering", AAAI Workshop on Knowledge-based Electronic Markets 2000 (KBEM'00). Austin, TX. July, 2000.

[Caruana and Freitag, 1994]

Caruana, R., Freitag, D., "Greedy Attribute Selection", Proceedings of the 11th International Conference on Machine Learning (ICML'94), pp. 26-28. 1994.

[Carroll and Rosson, 1987]

Carroll, J., Rosson, M. B., "The Paradox of the Active User". In J. M. Carroll (Ed.), Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. Cambridge, MA: MIT Press. 1987.

[Castelfranchi and Falcone, 1998]

Castelfranchi, C., Falcone, R., "Principles of Trust for MAS: Cognitive Anatomy, Social Importance, and Quantification". In: Demazeau, Y. (ed.), Proceedings of the Third International Conference on Multi-Agent Systems, IEEE Computer Society, pp. 72-79, Los

Alamitos, 1998.

[CDNow]
CDNow.com – http://www.cdnow.com

[Chatterjee et al., 1998]
Chatterjee, P., Hoffman, D.L., Novak, T.P., "Modeling the Clickstream: Implications for Web-Based Advertising Efforts", a Working Paper, Vandetbilt University, May 1998.

[Chen and Sycara, 1998]
Chen, L., Sycara, K.: "WebMate: A Personal Agent for Browsing and Searching," Proceedings of the 2nd International Conference on Autonomous Agents and Multi Agent Systems, AGENTS '98, ACM, pp. 132-139, May 1998.

[Chen et all, 2000]
Chen, Z., Meng, X., Zhu, B., Fowler, R., "Websail: From on-line learning to web search". Proceedings of the 2000 International Conference on Web Information Systems Engineering, June 2000.

[Clark and Nibblet, 1989]
Clark, P., Niblett, T., "The CN2 Induction Algorithm", Machine Learning, vol. 3, Kluwer Academic Publishers, The Netherlands, pp. 261-83, 1989.

[Cohen, 1995a]
Cohen, W., "Learning to classify english text with ILP method". In L. De Raedt, editor, Proceedings of the 5th International Workshop on Inductive Logic Programming, Scientific report, pages 3--24. Department of Computer Science, Katholieke Universiteit Leuven, 1995.

[Cohen, 1995b]
Cohen, W., "Fast Effective Rule Induction". In Proceedings of the 12th International Machine Learning Conference (ML95), pp. 115--123. San Francisco, CA:Morgan Kaufmann. 1995.

[Cohen and Singer, 1999]
Cohen, W. W., Singer, Y., "A Simple, Fast, and Effective Rule Learner". Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), pp. 335-342, October 1999

[Cooley et al., 1999]
Cooley, R., Tan, P.-N., Srivastava, J. "Websift: the Web site information filter system". In Proceedings of the 1999 KDD Workshop on Web Mining, San Diego, CA. Springer-Verlag, in press. 1999.

[Cost and Salzberg, 1993]
Cost, S., Salzberg, S.," A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features". Machine Learning vol.10, pp.57-78. 1993.

[Cunningham et al., 2001]
Cunningham, P., Bergmann, R., Schmitt, S., Traphoner, R., Breen, S., Smyth, B., "WEBSELL: Intelligent Sales Assistants for the World Wide Web". To be presented at e-Business and e-Work 2001 (e-2001). October 2001.

[Dastani et al., 2000]

Dastani, M., Jacobs, N., Jonker, C. M., Treur, J., "Modelling User Preferences and ediating Agents in Electronic Commerce", Proceedings of the Twelfth Belgium-Netherlands Artificial Intelligence Conference (BNAIC'00), pp. 297--298, 2000.

[de la Rosa et al., 1994]

de la Rosa J. Ll., Serra I., and Aguilar-Martin J., "Heuristics for Co-operation of Expert Systems. Application to Process Control", Ed. PIAR, ISBN 84-605-0275-9, 1994.

[de la Rosa et al., 1999]

de la Rosa J.Ll., Garcia R., Innocenti B., Muñoz I., Figueras A., Ramon J.A., Montaner M., "Rational Dynamical Physical Agents", 3rd Robocup Workshop, Vol. on RoboCup-99: Robot Soccer World Cup III, Lecture Notes in AI Nº1395, Eds. Springer-Verlag, Estockholm (Sweden). July 31-August 6, 1999.

[Deerweester et al., 1990]

Deerweester, S., Dumais, S., Furnas, G., Landuer, T., Harshman, R., "Indexing by Latent Semantic Analysis", Journal of the American Society for Information Science, Vol, 41, No. 1, pp. 391-407. 1990.

[Devine et al., 1997]

Devine, P., Paton, R., Amos, M., "Adaptation of Evolutionary Agents in Computational Ecologies", BCEC 97, Sweden., 1997.

[Duda and Hart, 1973]

Duda, R., Hart, P.: "Pattern Classification and Scene Analysis", John Wiley & Sons, New York, ISBN-0471223611, 1973.

[Elofson, 1998]

Elofson, G., "Developing Trust with Intelligent Agents: An Exploratory Study", In Proceedings of the First International Workshop on Trust, pp. 125-139. 1998.

[Etzioni, 1996]

Etzioni, O., "The world--wide web: Quagmire or gold mine?", Communications of the ACM, vol. 39, pp. 65--68, November 1996.

[Euroagri-Indrycha]

EUREKA PROJECT E!2146 EUROAGRI INDRYCHA: "Design of an intelligent drying chamber (IDC) for the curing of hams and sausages". http://www3.eureka.be/Home/projectdb/PrjFormFrame.asp?pr_id=2146

[Francis and Ram, 1993]

Francis, A. G., Ram, A., "The Utility Problem in Case-Based Reasoning". CaseBased Reasoning: Papers from the 1993 Workshop. AAAI Press (WS-93-01). Washington. 1993.

[Foltz, 1990]

Foltz, P. W., "Using Latent Semantic Indexing for Information Filtering", Proceegings of the ACM Conference on Office Information Systems, ACM/SIGOIS, pp. 40-47. New York, 1990.

[Gambetta, 1990]

D. Gambetta, "Can We Trust Trust?". In, Trust: Making and Breaking Cooperative

Relations, Gambetta, D (editor). Basil Blackwell, pp. 213-237. Oxford. 1990.

[Goldberg et al., 1992]

Goldberg D., Nichols D., Oki B. M., Terry D., "Using Collaborative Filtering to Weave an Information Tapestry", Communications of the ACM, Vol. 35, No. 12, pp. 61-70, December 1992

[Good et al., 1999]

Good, N., Schafer, J., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., Riedl, J., "Combining collaborative filtering with personal agents for better recommendations", In Proceedings of AAAI, pages 439--446. AAAI Press, 1999.

[Greening, 1997]

Greening D., "Building Consumer Trust with Accurate Product Recommendations.", Likeminds White Paper LMWSWP-210-6966, 1997.

[Hayes and Cunningham, 1999]

Hayes, C. Cunningham, P., "Smart Radio - a proposal". Trinity College Dublin, Computer Science, Technical Report, TCD-CS-1999-24. April 1999.

[Hayes and Cunningham, 2000]

Hayes, C., Cunningham, P., "Smart Radio: Building Music Radio on the Fly". In Proceedings of Expert Systems 2000 (ES2000), Cambridge, UK, December 2000.

[Hayes et al., 2001]

Hayes, C., Cunningham, P., Smyth, B., "A Case-Based Reasoning View of Automated Collaborative Filtering". Trinity College Dublin, Computer Science, Technical Report, TCD-CS-2001-09. March 2001.

[Herlocker et al., 2000]

Herlocker, J., Konstan, J., and Riedl, J., "Explaining Collaborative Filtering Recommendations". To be presented at ACM 2000 Conference on Computer Supported Cooperative Work , December 2-6, 2000.

[Herlocker et al., 1999]

Herlocker, J., Konstan, J., Borchers, A., Riedl, J. "An Algorithmic Framework for Performing Collaborative Filtering". To appear in Proceedings of the 1999 Conference on Research and Development in Information Retrieval. August 1999.

[Hill et al., 1995]

Hill, W., Stead, L., Rosenstein, M., Furnas, G., "Recommending and evaluating choices in a virtual community of use", Proceedings of the Conference on Human Factors in Computing Systems (CHI'95), pp. 194-201, Denver, CO, ACM Press, 1995.

[Hofmann and Puzicha, 1999]

Hofmann, T., Puzicha, J.: "Latent Class Models for Collaborative Filtering", the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99), Stockholm, ISBN 1-55860-613-0, pp. 688-693, July 31 - August 6, 1999.

[Holte and Yan, 1996]

Holte, R. C., Yan, N. Y. "Inferring what a user is not interested in". In AAAI Spring Symp. on Machine Learning in Information Access, Stanford, USA, 1996.

[Huberman and Kaminsky, 1996]

Huberman, B., Kaminsky, M.: "Beehive: A System for Cooperative Filtering and Sharing of Information". Technical report, Dynamics os Computation Group, Xerox, Palo Alto Research Center, Palo Alto, CA. 1996.

[Jennings and Higuchi, 1993]

Jennings, A., Higuchi, H., "A User Model Neural Network for a Personal News Service". User Modeling and User-Adapted Interaction 3:1-25. 1993.

[Jensen, 1996]

Jensen, F.V.: "An Introduction to Bayesian Networks". New York: Springer, 1996.

[Joachims et al., 1997]

Joachims, T., Freitag, D., Mitchell, T.: "WebWatcher: A Tour Guide for the World Wide Web". In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence. Nagoya, Japan, pp. 770--775, 1997.

[Jonker and Treur, 1999]

Jonker, C. M., Treur, J., "Formal Analysis of Models for the Dynamics of Trust based on Experiences". In: F. J. Garijo, M. Boman (eds.), Multi-Agent System Engineering, Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW'99. Lecture Notes in AI, vol. 1647, pp. 221-232, Springer Verlag, Berlin, 1999

[Kamba et al., 1995]

Kamba, T., Bharat, K., Albers, M. C., "The Krakatoa Chronicle--An Interactive, Personalized, Newspaper on the Web" in Proc. of the Fourth International World Wide Web Conference, pp. 159-170, November 1995.

[Kibler and Aha, 1988]

Kibler, D., Aha, D. W., "Case-based classification". In Proceedings of the Case-Based Reasoning Workshop at AAAI 1988 (pp. 62-67). 1988.

[Kobsa et al., 2001]

Kobsa, A., Koenemann, J., Pohl, W., "Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships". The Knowledge Engineering Review, forthcoming. 2001.

[Konstan et al., 1997]

Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Riedl, J., "GroupLens: Applying Collaborative Filtering to Usenet News. Communications of the ACM 40, 3, pp. 77-87. 1997.

[Kowalski, 1997]

Kowalski, G., "Information Retrieval Systems - Theory and Implementation", Kluwer Academic Publishers, ISBN-0-7923-9926-9. Norwell, MA, 1997.

[Koychev, 2000]

Koychev, I., "Gradual Forgetting for Adaptation to Concept Drift". In Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning, Berlin. 2000.

[Krulwich, 1997]

Krulwich, B. "LIFESTYLE FINDER: Intelligent User Profiling Using Large-Scale Demographic Data". In AI
Magazine. 18, 2. 37-45. Summer 1997.

[Krulwich and Burkey, 1995]
Krulwich, B. and Burkey, C.: "ContactFinder: Extracting indications of expertise and answering questions with referrals". In: The working Notes of the 1995 Fall Symposium on Intelligent Knowledge Navigation and retrieval. Technical Report FS-95-03, The AAAI Press, pp. 85 -- 91. 1995.

[Krulwich and Burkey, 1996]
Krulwich, B., Burkey, C. "Learning user information interests through extraction of semantically significant phrases". In Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access. Stanford, CA, March 1996.

[Lang, 1995]
Lang, K., "NewsWeeder: Learning to filter news". Proceedings of the Twelfth International Conference on Machine Learning (pp. 331--339). Lake Tahoe, CA, 1995.

[Leake, 1996]
Leake, D., "Case-based Reasoning: Experiences, Lessons, and Future Directions", AAAI Press/MIT Press, ISBN 0-262-62110-X. 1996.

[Leake and Wilson, 1998]
Leake, D.B., Wilson, D.C., "Categorizing Case-Base Maintenance: Dimensions and Directions". Advances in Case-Based Reasoning: 4th European Workshop, EWCBR-98, Dublin (Ireland). September 1998.

[Lewis and Gale, 1994]
Lewis, D. D., and Gale, W. A., "A sequential algorithm for training text classifiers". In Proceedings of SIGIR94, 17th ACM International Conference on Research and Development in Information Retrieval, 3--12. 1994.

[Lieberman, 1995]
Lieberman, H., "Letizia: An Agent That Assists Web Browsing". In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95), pages 924--929, Montreal, August 1995.

[Lieberman et al., 1999]
Lieberman, H., Van Dyke, N. W., Vivacqua, A. S., "Let's Browse: A Collaborative Web Browsing Agent", in proceedings of International Conference on Intelligent User Interfaces, Redondo Beach, CA, 1999.

[Maes, 1994]
Maes, P., "Agents that reduce work and information overload" Communications of the ACM vol. 37 no. 7, pp 30-40, July 1994.

[Malone et al., 1987]
Malone, T., Grant, K., Turbak, F., Brobst, S., Cohen, M.. Intelligent InformationSharing Systems. Communications of the ACM 30 (5), 390--402. 1987.

[Maloof and Michalski, 2000]

Maloof, M. A., Michalski, R. S., "Selecting examples for partial memory learning". Machine Learning, 41:27--52, 2000.

[Marsh, 1994]

Marsh, S. P., "Formalising Trust as a Computational Concept". Phd Thesis, Department of Computing Science and Mathematics, University of Stirling, 1994.

[Meléndez et al., 2000]

Meléndez, J., de la Rosa, J.L., Macaya, D., Colomer, J., "Case Based Approach for generation of recipes in Batch Process Control", CCIA2000 Congrés Català d'Intel.ligència Artificial, Vilanova i La Geltrú, 2000.

[Meléndez et al., 2001]

Meléndez, J. Colomer, J, Macaya, D., "Case Based Reasoning methodology for supervision", Accepted in the european control conference (ECC'01), Oporto, (Portugal). September, 2001

[Minio and Tasso, 1996]

Minio, M, Tasso, C. "User Modeling for Information Filtering on INTERNET Services: Exploiting an Extended Version of the UMT Shell". UM96 Workshop on "User Modeling for Information Filtering on the WWW", Kailua-Kona, Hawaii, USA, January, 2-5, 1996.

[Mitchell, 1996]

Mitchell M.: "An Introduction to Genetic Algorithms". A Bradford Book. The MIT Press, ISBN 0-262-63185-7. Cambridge, Massachusetts, 1996.

[Mitchell, 2000]

Mitchell M., "Life and Evolution in Computers", Darwinian Evolution Across the Disciplines, 2000.

[Mitchell et al., 1994]

Mitchell T., Caruana R., Freitag D., McDermott, J., Zabowski D., "Experience with a Learning Personal Assistant". Communications of the ACM 37.7 81-91. 1994.

[Mitchell et al., 1985]

Mitchell, T. M., Mahadevan, S., Steinberg, L.: "LEAP: A learning apprentice for VLSI design". In Proceedings of the Ninth International Joint Conference on Artificial Intelligence, pages 573--580, Los Altos, California. Morgan Kaufmann. 1985.

[Mladenic, 1996]

Mladenic, D.: "Personal WebWatcher: Implementation and Design". Technical Report IJS-DP-7472, Department of Intelligent Systems, J.Stefan Institute, Slovenia. 1996.

[Mobasher et al., 2000]

Mobasher, B., Cooley, R., Srivastava, J.: "Automatic Personalization Based on Web Usage Mining", Communications of the ACM (CACM), Vol. 43, No. 8, August 2000.

[Montaner et al., 2001]

Montaner, M., de la Rosa, J.Ll., Batlle, E., Guixeras, B., "GenialChef: un Recomanador de Restaurants Personalitzat". Forthcoming. 2001.

[Morita and Shinoda, 1994]

Morita, M., Shinoda, Y., "Information filtering based on user behaviour analysis and best match text retrieval", Proceedings of the 17th ACM Annual International Conference on Research and Development in Information Retrieval (SIGIR'94), Dublin, Ireland, Springer-Verlag, 272-81. 1994.

[Moukas, 1997]

Moukas, A.: "Amalthaea: Information Filtering and Discovery using a Multiagent Evolving System". Journal of Applied AI, Vol. 11, No. 5, pp. 437-457, 1997.

[Nichols, 1997]

Nichols, D.M., "Implicit Rating and Filtering", Proceedings of 5 th DELOS Workshop on Filtering and Collaborative Filtering, Budapest, Hungary, 10-12 November, ERCIM 31-36. 1997.

[Oard and Marchionini, 1996]

Oard, D.W., Marchionini, G., "A Conceptual Framework for Text Filtering", Technical Report CARTR -830, Human Computer Interaction Laboratory, University of Maryland at College Park. 1996.

[Oller et al., 2000]

Oller A., de la Rosa J.Ll., García R, Ramon JA, Figueras A. "Micro-Robots playing soccer games: a real implementation based on a multi-agent decision-making structure", Int. Journal of Intelligent Automation and Soft Computing. Special Issue on Soccer Robotics: MIROSOT '97, Vol. 6, No. 1, January 2000.

[Ontañon and Plaza, 2001]

Ontañon, S., Plaza, E., “Collaboration Policies for Case-Based Reasoning Agents”. In Proc. Workshop on Learning Agents Autonomous (Agents'2001). Montreal. May, 2001.

[Orwant, 1995]

Orwant, L. J.: "Heterogeneous Learning in the Doppelganger User Modelling System", in User Modelling and User Adapted Interaction , 4(2), pp: 107-130. 1995.

[Pazzani, 1999]

Pazzani, M., "A Framework for Collaborative, ContentBased and Demographic Filtering". Artificial Intelligence Review, 1999.

[Pazzani and Billsus, 1997]

Pazzani, M., Billsus, D., "Learning and Revising User Profiles: The Identification of Interesting Web Sites", Machine Learning 27, Kluwer Academic Publishers, pp. 313---331, 1997.

[Pazzani et al., 1996]

Pazzani, M., Muramatsu, J., Billsus, D.: "Syskill & Webert: Identifying interesting web sites". In: Proceedings of the Thirteenth National Conference on Artificial Intelligence. Portland, OR, pp. 54--61. 1996.

[Potter and Trueblood, 1988]

Potter, G., Trueblood, R., "Traditional, Semantic, and Hyper-Semantic Approaches to Data Modeling". IEEE Computer, 21(6):53-63, June 1988.

[Pretschner and Gauch, 1999]

Pretschner, A., Gauch, S.: "Ontology Based Personalized Search". Proc. 11th IEEE Intl. Conf. on Tools with Artificial Intelligence (ICTAI'99), pp. 391-398, Chicago, IL, November 1999.

[Quinlan, 1983]

Quinlan, J.R.: "Learning efficient classification procedures and their application to chess end games". In Machine learning: an artificial intelligence approach, edited by R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, 463--482. Tioga, Palo Alto, CA. 1983.

[Quinlan, 1994]

Quinlan J.R.: "The Minimum Description Length Principle and Categorical Theories", in Cohen W.W. and Hirsh H.(eds.), Machine Learning: Proceedings of the Eleventh International Conference (ML94), Morgan Kaufmann, San Mateo, CA, 1994.

[Resnick et al., 1994]

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J. "Grouplens: An open architecture for collaborative filtering of netnews", In Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work, Sharing Information and Creating Meaning, pages 175-186, 1994.

[Rich, 1979]

Rich, E.: "User Modeling via Stereotypes". Cognitive Science, 3:329--354, 1979.

[Riordan and Sorensen, 1995]

Riordan, A., Sorensen, H.: "An intelligent agent for high-precision information filtering". In Proceedings of the CIKM-95 Conference, 1995.

[Sabater and Sierra, 2000]

Sabater, J., Sierra, C., “REGRET: A reputation model for gregarious societies”. Research Report. Institut d’Investigació i Intel.ligència Artificial. October 2000.

[Sakagami et al., 1997]

Sakagami, H., Kamba, T., Koseki, Y.: "Learning personal preferences on online newspaper articles for user behaviors", in: Proc. 6th Int. World Wide Web Conference, pp. 291-300. 1997.

[Salton and Buckley, 1988]

Salton, G., Buckley, C., "Term-Weighting Approaches in Automatic Text Retrieval", Information Processing and Management, 24(5), 513-523. 1988.

[Salton and Buckley, 1990]

Salton, G., Buckley, C. "Improving Retrieval Performance by Relevance Feedback". In Spark Jones and Willet, (Eds.) Readings in Information Retrieval. San Francisco, CA: Morgan Kauffman, 1990.

[Salton and McGill, 1983]

Salton, G., McGill, M.: "Introduction to Modern Information Retrieval". McGraw-Hill Publishing Company, New York, NY, 1983.

[Sangüesa and Cortés, 2000]

Sangüesa, R., Cortés, U., “Workshop on Agent-Based Recommender Systems”. Workshop 9. The Fourth International Conference on Autonomous Agents 2000. June 3-7. Barcelona

(Spain). 2000.

[Sarwar et al., 2000]

Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. "Analysis of Recommender Algorithms for E-Commerce". To be presented at the ACM E-Commerce 2000 Conference. Oct. 17-20, 2000.

[Sarwar et al., 1998]

Sarwar, B.M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B. and Riedl, J., "Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System". In Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW'98), pp. 345-354. November, 1998.

[Schafer et al., 2001]

Schafer, J.B., Konstan, J., Riedl, J., "Electronic Commerce Recommender Applications". Journal of Data Mining and Knowledge Discovery, vol. 5 nos. 1/2, pp 115-152. January, 2001.

[Schillo et al., 2000]

Schillo, M., Funk, P., Rovatsos, M., "Using trust for detecting deceitful agents in artificial societites". In Applied Artificial Intelligence, Special Issue on Trust, Deception and Fraud in Agent Societies, 2000.

[Schwab et al., 2000]

Schwab I., Kobsa A., Koychev I., "Learning about Users from Observation", AAAI 2000 Spring Symposium: Adaptive User Interface. 2000.

[Schwab et al., 2001]

Schwab, I., Kobsa, A., Koychev, I.: "Learning User's interests through Positive Examples Using Content Analysis and Collaborative Filtering". Submitted. 2001.

[Shardanand, 1994]

Shardanand, U., "Social Information Filtering for Music Recommendation", MIT EECS M. Eng. thesis, also TR-94-04, Learning and Common Sense Group, MIT Media Laboratory, 1994.

[Shardanand and Maes, 1995]

Shardanand, U., Maes, P.: "Social information filtering: algorithms for automating `word of mouth'" in Conf. proc. on Human factors in computing systems (CHI'95), pp. 210-217, Denver, CO USA, 1995.

[Sheth, 1994]

Sheth, B.: "A Learning Approach to Personalized Information Filtering", M.S. Thesis, Massachusetts Institute of Technology, 1994.

[Sheth and Maes, 1993]

Sheth, B., Maes, P.: "Evolving agents for personalitzed information filtering". In Proceedings of the Ninth Conferece on Artificial Intelligence for Applications. IEEE Computer Society Press, 1993.

[Steels and Vogt, 1997]

Steels L., Vogt P., "Grounding adaptive language games in robotic agents", Proceedings of

the Fourth European Conference on Artificial Life, pp. 473-484, Brighton, U.K., July 1997.

[Sorensen et al., 1997]

Sorensen H., Riordan A. O., Riordan C. O., "Profiling with the INFOrmer Text Filtering Agent", Journal of Universal Computer Science, Vol. 3, No. 8, pp. 988-1006. 1997.

[Sorensen and McElligot, 1995]

Sorensen, H., McElligot, M.: "PSUN: A Profiling System for Usenet News". CKIM '95 Workshop on Intelligent Information Agents, Baltimore, Maryland, December.1995.

[Stefani and Strappavara, 1998]

Stefani, A., Strappavara, C.: "Personalizing access to web sites: The SiteIF project". In Proc. 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98, Pittsburgh, USA, June 1998.

[Teerven and Hill, 2001]

Terveen, L.G., Hill, W., "Beyond Recommender Systems: Helping People Help Each Other", in Carroll, J. (ed.), HCI in the New Millennium, Addison Wesley. 2001.

[Torra, 2001]

Torra, V., “On the integration of numerical information: from the arithmetic mean to fuzzy integrals”. Torra V. (Ed). Information fusion in data mining. Physiin-Verlag. (Forthcoming). 2001.

[Watson, 1997]

Watson, I., "Applying Case-Based Reasoning: Techniques for Enterprise Systems". Published by Morgan Kaufman Publishers. ISBN: 1-55860-462-6. July, 1997

[Webb and Kuzmycz, 1996]

Webb, G., Kuzmycz, M., "Feature Based Modelling: A Methodology for Producing Coherent, Consistent, Dynamically Changing Models of Agents' Competencies", User Modelling and User-Adapted Interaction, Vol. 5, pp 117-150. 1996.

[Widmer and Kubat, 1996]

Widmer G., Kubat M., "Learning in the presence of concept drift and hidden contexts". Machine Learning 23: 69--101, Kluwer Academic Publishers. 1996.

[Wolf et al., 1999]

Wolf, J., Aggarwal, C., Wu, K., Yu, P. "Horting hatches and egg: A new graph-theoretic approach to collaborative filtering". In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 1999.

[Wooldridge, 1999]

Wooldridge, M., "Intelligent Agents". Multiagent Systems. Editor G. Weiss, The MIT Press, April 1999.

[Wooldridge and Jennings, 1995]

Wooldridge, M., Jennings, N.R., "Intelligent Agents: Theory and Practice", Knowledge Engineering Review, Vol 10(2), pp. 115-152, 1995.

[Wu and Sun, 2001]

Wu, D., Sun, Y., "The Emergence of Trust in Multi-Agent Bidding: A Computational

Approach". Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34. Hawaii. January, 2001.

[Yan and Garcia-Molina, 1995]

Yan, T.W., Garcia-Molina, H.: "SIFT -- A Tool for Wide-Area Information Dissemination". Proceedings of the 1195 USENIX Technical Conference, pp. 177-186, 1995.

[Yao, 1995]

Yao, Y. Y., "Measuring Retrieval Effectiveness Based on User Preference of Documents", Journal of the American Society for Information Science, 46(2), pp. 133-145, 1995.

[Yu and Singh, 2000]

Yu, B., Singh, M. P., "A social mechanism of reputation management in electronic communities". In cooperative information agents, CIA-2000, pages 154-165. Boston, MA, USA. 2000.

[Zacharia, 1999]

Zacharia, G., "Collaborative reputation mechanisms for online communities". Master's thesis, Massachusetts Institute of Technology. September, 1999.