MASHA: A Multi Agent System Handling User and Device Adaptivity of Web Sites †

D. Rosaci and G.M.L. Sarné
DIMET, Università "Mediterranea" di Reggio Calabria, Loc. Feo di Vito, 89060 Reggio Calabria (Italy)
E-mail: {domenico.rosaci,sarne}@unirc.it

October 5, 2005

Abstract. A user that navigates on the Web using different devices should be characterized by a global profile, which represents his behaviour when using all these devices. Then, the user's profile could be usefully exploited when interacting with a site agent that is able to provide useful recommendations on the basis of the user's interests, on one hand, and to adapt the site presentation to the device currently exploited by the user, on the other hand. However, it is not suitable to construct such a global profile by a software running on the exploited device since this device (e.g., a mobile phone or a palmtop) may have limited resources. Therefore, in this paper, we propose a multi agent architecture, called MASHA, handling user and device adaptivity of Web sites, in which each device is provided with a client agent that autonomously collects information about the user's behaviour associated to just that device. However, the user profile contained in this client is continuously updated with information coming from a unique server agent, associated with the user. Such information is collected by the server agent from the different devices exploited by the user, and represents a global user profile. The third component of this architecture, called *adapter agent*, is capable to generate a personalized representation of the Web site, containing some useful recommendations derived by both an analysis of the user profile and the suggestions coming from other users exploiting the same device.

Keywords: Information Agents, Recommender Systems, Web Adaptivity, Device Adaptivity

1. Introduction

Nowadays, the large development of the Internet has involved a large number of potential users and an overwhelming amount of Web sites that need to effectively interact with their visitors. In this context, it is also necessary to consider a further important factor relative to the quick evolution of electronic and telecommunication technologies. Besides of using desktop personal computers, users can connect to the Internet with notebooks, cell phones, palmtops and wireless Personal

Digital Assistants to navigate on the Web. Each of these devices has its own physique and technological characteristics (such as display or bandwidth capabilities) and usually it is vastly different from its desktop counterpart; nonetheless providers deliver the same content to all device typologies (Resnick and Varian, 1997). Many applications are available for supporting the navigation through an interesting site. The most common of these applications are based on *recommender systems* (Sarwar et al., 2000; Shafer et al., 2001) and on management of user profiles (usually rough) which are exploited to propose personalized sites. A recommender system, in order to perform its tasks, must process available data; this can be done by exploiting various computational tools such as lists and information filters. Some recommender systems, called *Content-based*, recommend to a user the resources appearing to be the most similar to those he has already accessed in the past. Other systems, called *Collaborative Filtering*, recognize commonalities among users on the basis of their interests and generate recommendations taking into account inter-user comparisons. However, in spite of recommender systems and user profile managers, when a user accesses a site, in general he must personally search the issues of his interest through the site. We argue that, for improving the effectiveness of the service, it is necessary to increase the interaction between the site and the user, on one hand, and to construct a personal profile of the user, taking into account his desires, interests and behaviors, on the other hand. The solution of the problems previously outlined, requires some challenges to be tackled. First, a user can access *many* sites; a faithful and complete profile of him can be constructed only by taking into account his behavior when accessing all these sites. In other words, it should be possible to construct a *unique structure*, storing his profile and, therefore, representing his behavior on accessing *all these sites*. Second, for a given user and site, it should be possible to compare the profile of the user with the profile of the site and then extract the features that probably in the future will be of interest to the user. In the past, various approaches have been proposed for handling surfing activities. Some of them are based on traditional recommender systems (Resnick and Varian, 1997); other ones (Guttman et al., 1998; Lau et al., 2000; Ardissono et al., 2001; Garcia et al., 2002) have been designed in the context of the agent technology. These approaches construct, maintain and exploit a user profile; therefore, we can consider them adaptive with respect to the user; however, to the best of our knowledge, none of them is adaptive with respect to the device. The necessity that recommender systems take into account, when they give their suggestions, also the typology of the exploited device, is an emerging issue in designing these systems. If a user accesses a site with a desktop

PC, the site manager can propose its recommendations by using a presentation having a high structural complexity, supposing the user can handle such a presentation by using its browser equipped with full functionalities. Instead, when the exploited device is a cellular phone, the site presentation has to be lighter than in the previous case. Note that the two aspects of generating suggestions and choosing the most suitable site presentation for the exploited device are not orthogonal. A user exploiting a cellular phone has to be provided with suggestions that take into consideration the used device; as an example of collaborative filtering method, in this case, the suggestions of the other users exploiting the same devices should be preferred when generating recommendations. This way, the adaptivity with respect to device does not involve only graphical aspects of the presentation. Finally, we also note that the interest of a user in a particular concept (e.g. *seaside places*) is a characteristic of the user that he shows regardless the device he exploits. However, if we measure the user interest by a metric, we argue that the value that measures the interest depends on the exploited device. For example, if the user visits a travel agency site accessing for ten minutes an instance of the concept *seaside places* by using a PC, we can derive a determined interest value for this concept; but this same value is significantly smaller than the first one related to the same user that visits the same concept for the same amount of time by using a cellular phone, that implies a higher cost. For further clarifying this issue, suppose that the instance of *seaside places* in the site above is a video presentation, onerous to be accessed by the user that is exploiting a cellular phone: although the user is interested in seaside places, he might not have access to the concept instance above.

In various computer science research fields, a large variety of approaches adapting their behavior to the device that the user is exploiting, has been proposed (Anderson et al., 2001; Ardissono et al, 2003; De Bra et al., 2002). However, these approaches have not been specifically designed for supporting Web navigation. It should be suitable to propose analogous techniques that address this important issue. Software agents seem a promising solution in this context. A software agent is a computing entity capable of perceiving dynamic changes in the environment and, consequently, of autonomously performing user delegated tasks. This work presents a multi agent system called *Multi Agent System Handling Adaptivity* (MASHA), in which each Web user is monitored by a software agent, provided with a personal profile, that supports his navigation. Each Web site manager is also supported by an agent able to interact with the agents of the visitors to adapt site presentation to users' characteristics and to generate some useful suggestions. The idea underlying our approach is the following. A user that

navigates on the Web using different devices should be characterized by a global profile that takes into account his behaviour when using all such devices. However, it is not suitable, in our opinion, to construct such a global profile based on a software running on the exploited device, since this device (e.g., a mobile phone or a palmtop) may have limited resources. Therefore, we choose to provide each device with a client agent that autonomously collects information about the user's behaviour associated to just that device. However, we think that such a client agent, when it has to interact with a Web site for allowing site adaptivity, can take relevant benefit if its local user's profile is updated with information coming from the other client agents associated with the same user. In our architecture, the information are collected in a global user's profile by a server agent, that is assumed to have more relevant resources than the client. Therefore, both client and server agent are autonomous when performing their activities, as we can see by observing that the client agent interacts with a Web site without the assistance of the server agent, while the server agent simply acts as a collector of available information coming from each client agent without explicitly depending on the presence of a particular client. However, client and server agent *collaborate* in order to mutually improve their knowledge. In an analogous way, the adapter agent also collaborates with the client agent to adapt the site presentation to the user profile; however, in any case, the adapter is able to independently generate the site presentation, even in absence of a collaboration with client agents. Many other agent-based approaches have been proposed in the literature for generating recommendations (Silvestri et al., 2004; Buccafurri et al., 2002; Garruzzo et al., 2002; Parsons et al., 2004). Some of them act as content-based recommender systems, other ones as collaborative filtering recommender systems. Other existing approaches are both content-based and collaborative filtering. Likewise, the MASHA system proposed in this paper shows both these two different capabilities. However, unlike the aforementioned approaches, MASHA presents three original characteristics, namely: (*i*) that of considering, in order to construct the global user profile, the different devices exploited by the user; (*ii*) that of exploiting two different agent types for building such a profile, called *client agent* and *server agent*, respectively. The former is associated to both each user and each single device exploited by that user, and it is able to manage only information deriving from the use of just that device; the latter is associated to the user and collects information coming from all the user's devices; (*iii*) that of generating the recommendations on the site-side by using a third agent type, called *adapter*, that performs this task. These characteristics generate three main advantages. Firstly, the global user's profile, that

is used by the recommendation algorithm in order to generate useful suggestions for the user visiting a site, effectively models the user's interest, since when an interest rate is assigned to a concept, this rate takes into account the exploited device. Secondly, the client agent, that runs on the device, performs a relatively light task and it is responsible of neither the construction of the whole user's profile, that is built by the more powerful server agent instead, nor the recommendations that are generated by the adapter agent of the site. Finally, the collaboration between client and server agent allows a periodical update of the profile stored in the client. This way, the client has the global profile available when it interacts with the site for receiving the recommendations. These advantages make MASHA able to achieve more effective results in the generated recommendations. In fact, both the content-based and the collaborative filtering analysis are conducted by the MASHA adapter, associated to a site, by interacting with the client agent associated to the device currently exploited by the visitor. This leads the adapter to consider, in the information filtering activity, only users that exploit the same device of this visitor. Moreover, since the client agent stores a global profile obtained by the associated server agent, the content-based activity can be performed with at least the same effectiveness of the other recommendation methods. In addition, also the content-based activity performed by using MASHA, instead of other recommender systems, produces better results since the global profile has been constructed by the server agent taking into account the different exploited devices, so it is more precise in modelling user interests. We experimentally evaluate MASHA by comparing it with other recent profile-based recommender system approaches, and we observe a significative improvement of the recommendation performances. It is worth pointing out that the interaction between the client agent, that assists the user, and the adapter agent, that supports the site, is cooperative. Users explicitly reveal to the site those concepts that best match with the site content, but do not give any other information to the site (e.g., what are the other concepts they are interested in); on the other hand, the adapter does not reveal to the client of a user any information about other users since, although the recommendations are generated by matching different user profiles, such a matching is internal to the adapter that shows only the result to the visitor, through concepts that might be interesting for him. In other words, the privacy of the users is protected, except for the necessary information that the client must send to the adapter to realize site adaptivity and recommendations. We note that an important problem generally arises in multi agent systems, represented by the agents' need to understand each other. Each user's profile contains references to objects of interest

for the user. But how do these concepts have to be expressed in order to guarantee the agents an effective mutual understanding? Such an understanding is also necessary when comparing two user profiles in order to generate collaborative filtering recommendations. A proposal for facing this issue is that of using a common *vocabulary*, that is, a set of concepts whose meanings are shared by all the agents. Section 2 provides an overview of the MASHA architecture. Related work is examined in Section 3. Some experiments are presented in Section 4. Finally, in Section 5, some conclusions are drawn.

2. The Multi Agent System Handling Adaptivity

In this section we describe the multi agent architecture that we propose. First of all, we introduce in Subsection 2.1 some preliminary notions relative to the representation of objects and categories of interest that we use in our framework, while Subsection 2.2 provides the details of the different components of the MASHA system.

2.1. Representation of Objects and Categories of Interest

The MASHA framework supports the construction of a community of information agents composed by two different agent categories $C1$ and $C2$. Each agent of $C1$ is associated with a human user that navigates on the Web, while each agent of $C2$ is associated with an Internet site containing Web pages. The agents of $C2$ have to represent in their Web pages different objects belonging to different categories. For instance, in the case of the site of a travel agency, agents handle as objects travel proposals that belong to different categories like *sea travels*, *mountain travels*, etc. In our framework, each category of interest (for instance, *sea travel*) is called *concept* and it is represented by a pair (n, d), where n is the (string) identifier of the concept, called *name* of the concept while d is a text describing the meanings of the concept, called *description*. Consequently, all the agents of the community share a common *vocabulary*; that is, a set of concepts where each concept is uniquely identified by its *name* and whose semantics, represented by its *description*, is the same for all the agents. Each actual object present on a Web page is thus considered as an *instance* of a concept. Obviously, each Web page can contain many different concept instances relative to different concepts. In the example of the travel agency site, a Web page of this site will contain different instances of the concept *sea travel* (e.g. *travel in Sicily*, *travel in the Black Sea*, etc.), many instances of the concept *Mountain Travel* (e.g. *travel in the Himalayas*, *travel in the Etna*, etc.)

and so on. Many examples of vocabularies have been proposed in the literature and practically exploited. As a relevant example, we remember the North America Industry Classification (NAICS) (NAICS, 2006), that is an official hierarchical industrial classification used in North America that exploits unified evaluation criteria. There are many ways of implementing a vocabulary. As an example, a simple possibility that we will use in the implementation of our system (see Section 4) is represented by the XML-Schema that gives the possibility of defining a concept by using the notion of *element* and the concept instances by using the XML element instances. It is worth of pointing out, to avoid confusion, that the client agent visiting a Web site, actually accesses concept instances, such as *travel in the Himalayas* or *travel in Sicily*. However, if a client agent accesses an instance of a concept c, we assume that he is accessing the concept c. In other words, we state that for accessing a concept it is necessary to access an instance of that concept. We say that a user accesses a concept instance ci (and consequently its associated concept c) contained in a Web page p if he clicks on a hyperlink to ci. This implies the access only to ci, and it does not mean that the user accesses all the other concept instances contained in p. The notion of concept allows us to define the other notion of *interest* in the concept that measures quantitatively the interest for that category of objects represented by the concept. In this perspective, all the profiles that we will describe (profile of the client, server and adapter agent) contain *concepts* and not concept instances, since the notion of interest is defined only for concepts. Concept instances appear only on the Web pages of the Internet sites and the accesses of the client agents to these instances are monitored to construct the agent profiles and to determine the interest values for the associated concepts. Moreover, as it will be described later, in order to determine collaborative filtering recommendations, it is necessary to compute a similarity value between two agent profiles. In this case, the similarity measure that we introduce is based on the interest in the concepts belonging to the agent profiles, therefore it does not involve in any case the instances of the concepts.

2.2. MASHA Architecture

In this section we present the architecture of the MASHA system, that we propose in order to support a user in his Web navigation activities by exploiting a user profile. Such a profile takes into account both the behavior shown by the user while visiting Web sites during his navigation sessions and the devices exploited during these visits (indeed, a user can access a site using different devices). By exploiting this profile, *MASHA* acts as a Recommender System, in the sense that the

site currently visited by the user is managed by a MASHA agent that is able to adapt the format and the content of the site presentation to both the user profile and the characteristics of the device currently exploited. In particular, this personalized presentation contains some suggestions, in the form of Web links to content that is of potential interest to the user, also taking into account the exploited device. More specifically, we realistically suppose that the time spent by a user in visiting Web pages containing a concept is a rough measure of the user's interest in it; in this way, the user's interest in the concept is strictly related to the characteristics of the exploited device and the navigation costs. As an example, consider a user visiting the same Web page in two different navigation sessions, and spending each time n seconds. Suppose that this user in the first access exploits a mobile phone characterized by a low bandwidth and a high connection cost, while during the second visit he uses a personal computer characterized by a high bandwidth and a low connection cost. It is possible to argue that the user's interest shown for the page in the former access is greater than the one he shows in the latter case. As a consequence, by normalizing the visit time of each concept, mainly with respect to the characteristics of the exploited device and the navigation costs, it is possible to obtain a global user profile able to take into account all user interests independently of the device used. Thus, as it will described later on, this user's profile on one hand is updated with the information collected during the user's navigation sessions, carried out through different devices, and on the other hand it is exploited by the site in the presence of each user's device, to adapt its presentation and to generate suitable recommendations. To realize its aims, *MASHA* uses three typologies of agents, described in detail below, that operate during each navigation session. The first two agent typologies, called respectively *MASHA Client* and *MASHA Server*, support directly the user in the construction of his personal profile; the third agent typology, called *MASHA Adapter*, allows a Web site to adapt its presentation based on both the device and the user's profile stored in the device of the visitor. Each agent is registered with a central unit called *MASHA Agency* that assigns an agent identifier (AID) to the agent and also provides a white page service.

2.3. The Client Agent

In the MASHA framework a client agent is associated with each user exploiting a fixed device. Consequently, a user could have more client agents associated with him if he exploits different devices. A client agent is automatically activated (resp. deactivated) when a Web session starts (resp. ends *per se* or for an explicit user's choice). During a Web session,

the client agent stores and handles system and device information, on one hand, and updates locally the user profile on the basis of the visited concepts on the other hand. Below, the data structure of a client agent and its behavior in supporting both profile construction and user's activities are briefly described. In our description, we consider a user (U) that exploits a fixed device (D_i) and the associated client agent (CA_i).

Client Data Structure. The data structure of a CA_i can be described by the pair (CP_i, UP_i) constituted by the *Client Profile* (CP_i) and the *User Profile* (UP_i) data. More in detail:

the **Client Profile** CP_i stores device configurable data, and it is expressed by a tuple of the form $\langle AID_i, MSSet_i, \rho, T_M, P, \psi, k\rangle$ where AID_i is the *Client Agent* identifier; $MSSet_i$ is the *Maximum Size Set*, containing in its turn the three parameters MST_i, MSA_i, MSV_i that express the maximum sizes (in Kbyte) of text, audio and video contents, respectively, that U desires to handle when using the device D_i; ρ is a vector containing three coefficients, namely ρ_1, ρ_2 and ρ_3, associated with three possible actions that the user can perform on a Web page, namely visiting, storing, or printing the page; T_M is an integer coefficient used to evaluate the user's interest in a concept instance; P is the *attenuation period* (i.e., since the interest for a no longer visited concept decreases with the time, P represents the number of days between two consecutive interest decreasing operations); ψ is a function that is used to determine the U's interests relative to concepts, and that generates a decrease of these interests each P days when the associated concepts are no longer accessed; finally, k is a parameter that is exploited by the client agent in its interaction with the adapter agent of each visited site (see Section 2.5), and that represents the number of interesting concepts belonging to the visited site that the user desires to be considered in the site presentation;

the **User's Profile** UP_i locally stores the personal profile, based on the whole U's sessions history, that is updated by observing the hyperlinks clicked by the user that is exploiting D_i. More in particular, UP_i is a set of tuples $\langle c, LU_i, IR_i\rangle$, each tuple associated with a concept c, where:

- c is a concept identifier, belonging to the common shared vocabulary stored in the Agency.
- LU_i (*Last Update*) is the date of the last update of the coefficient IR_i (see below).

– IR_i (*Interest Rate*) is a measure of the U's interest in the concept c using the device i. More generally, several approaches have been proposed in the literature to define the user's interest in a Web page. As an example, in (Chan, 2000), the authors propose to consider the time spent on the page, the number of page accesses, the page length, and a possible score that the user gives for expressing his interest. Also in (Parsons et al., 2004) the visiting time is the main parameter considered in the definition of interest. Other approaches, as (Garruzzo et al., 2002; Buccafurri et al., 2002), define more in particular a measure of the interest in one element of a Web page. All these approaches consider as main parameters for evaluating such a measure the number of accesses to the element and the time spent by the user in visiting the page containing the element. Likewise, we define the measure of interest in a concept c on the basis of the actual time T spent by U in visiting the page containing c. Therefore, we represent it by a coefficient belonging to the interval $[0, 1]$, and assume that the maximum value of this coefficient is reached when the time spent on the page containing c is greater than or equal to a threshold T_M. Moreover, in order to consider also the possibility that U performs some particular actions regarding the Web page containing the concept as, for instance, storing or printing or simply reading the page, we introduce a coefficient ρ_a for the time T, that weights the interest measure in correspondence to each particular action a. The user U can set a customized value, belonging to the interval $[0, 1]$ for ρ_a in correspondence to each action a, $a = 1, 2, 3$.

  Then, we decide to compute the interest rate IR_i in a concept c by assuming that each way U accesses an instance of c, then IR_i linearly increases with the visit time T and that this contribution is maximum if T is greater to or equal than a threshold T_M. Consequently, IR_i depends on the ratio $\frac{T}{T_M}$, that we decided to weight by the coefficient ρ_a to take into account the different actions that the users can perform on the concept instance of c. We have also decided to perform a linear updating of IR_i, in the sense that the new value of IR_i due to the U's visit is computed as the mean value between the previous value of IR_i and the current measure $\frac{T}{T_M} \cdot \rho_a$, where the ratio $\frac{T}{T_M}$ is substituted by the value 1 if $T > T_M$.

  More formally, at each new update, IR_i is computed as follows:

$$IR_i = \begin{cases} (IR_i + \frac{T}{T_M} \cdot \rho_a)/2 & \text{if } T \leq T_M \\ (IR_i + \rho_a)/2 & \text{elsewhere} \end{cases}$$

Furthermore, in order to give more importance to the most recent concept accesses rather than the older ones, we periodically use the function ψ that accepts as input the interest rate IR_i and the last update LU_i and that returns as output a new value of IR_i, suitably decreased with respect to the original one on the basis of the temporal distance from the last update LU.

Client Behaviour. The behavior of a CA_i consists of four main sub-behaviours. More specifically:

Setup - At this stage, CA_i is logged into the U's server agent that provides CA_i with an agent identifier.

Monitoring - An active CA_i supports U as follows: (*i*) CA_i monitors the U's Web navigation sessions in order to determine both the interesting concepts and the user's behaviour when accessing them. Note that when the user accesses a new concept, not yet occurring in UP_i, a new element should be added into UP_i; (*ii*) After each Web session, CA_i sends to its server agent, for each visited concept c, a tuple of data of the form $\langle c, LU_i, IR_i \rangle$; (*iii*) The user's profile UP_i is periodically updated by the server agent. This update is a very relevant stage in our system, since it consists in a strong interaction between the client agent and its associated server agent, in which the server agent periodically communicates to the client agent the *current global value* of the interest rate IR of each concept. In fact, as it will be illustrated later on, the server agent of the user U stores a global IR for each concept, computed on the basis of all the different local IR_i values coming from each of the client agent CA_i. Therefore, the global IR represents the interest shown by the user U for the associated concept, taking into account *all the devices* exploited by U in the past. As a consequence, the updating stage consists of two different phases. In the first phase the client agent sends to its server agent, for each concept present in its profile, the current value of the associated interest rate. Then, in the second phase, the client agent receives from its server agent the corresponding global value of the interest rate.

Support - When U visits a Web site (W), the associated *Adapter Agent AA* (see Section 2.5), sends a suitable representation of the site content to CA_i, that selects from it all concepts that also belong to its user profile UP_i; these concepts are ordered

on the basis of the U's interest rate and returned to the adapter agent. This interaction between client agent and adapter agent thus allows the adapter agent that monitors the Web site W to know the interests of the user U for concepts belonging to the site, and then to suitably adapt the site presentation. Note that U explicitly allows the client agent to send the interest rates for the various concepts to the adapter agent. This behaviour configures a *collaborative scenario*in which users and Web managers explicitly cooperate by using a common protocol and a common vocabulary, to reach mutual advantages. This leads the user U to partially renounce the privacy of his interests, that are communicated to the adapter agent. However, this renunciation is relative only to those concepts that are common to both user and site, that is U does not communicate to W any information about his interests relative to other concepts that do not belong to W and that are not necessary for making effective the site adaptivity. Another important information that the client agent CA_i transmits to the adapter agent AA is the one relative to the exploited device, contained in the client profile of CA_i, that is the maximum size of text, audio and video that U wants or can handle on the device currently exploited. By using this information, AA is now able to generate a personalized presentation for U also taking into account the current device, as it is described in detail in Section 2.5.

Interest Decreasing - Each P days, CA_i performs for each interest rate IR_i associated to each concept c the update $IR_i = \psi(IR_i, LU_i)$, in order to take into account the "age" of the interest rate coefficient.

2.4. The Server Agent

The second agent typology, called *MASHA Server Agent*, is associated with a user and stores system and devices information, on one hand, and supports the user in its Web navigation sessions on the other hand. A server agent is always active and collects for each user's client agent the information about the concepts visited during the user Web activities. Then the server agent makes such personal information available at all user's client agents to construct their local user's profile that will be then exploited in the user activities support. This is an important feature of our agent system, since the client agents are able to live on associated devices that might have limited computation and storage capability; the assistance of the server agent, that runs on a more equipped machine (e.g. a personal computer), is fundamental for providing the user with an off-line collector of all the information obtained

by the different client agents that have assisted him in his navigation history. Below, the data structure of the server agent and the server agent's behavior when supporting both the user profile construction and the user's activities, are described. Without loss of generality we consider a user (U) that exploits a server agent (SA).

Server Data Structure. The data structure of SA is described by the pair (SP, GUP) where:

the **Server Profile** SP is represented by a tuple of the form $\langle AID, n, m, DP, f\rangle$ where AID is the *Server Agent* identifier; n is the number of client agents associated to the server agent; m represents the number of parameters necessary for computing the global interest rates of the various concepts (see below), DP is a matrix having n rows and m columns, where each DP_{ij} is the j-th parameter associated to the i-th device. This parameter is necessary, as we will describe below, for computing the contribution of CA_i to the global interest of a concept. It is possible to use as DP_{ij} parameter several characteristics of the connection associated to the device i, for instance the price per byte transmitted, the exploited bandwidth etc. The function f is used for computing the contribution of a client agent CA_i to the global interest of a concept. The function f accepts as input parameters a row of the matrix DP;

the **Global User Profile** GUP stores a global representation of U's interests relative to concepts present in his profile, as shown in his navigation behavior, exploiting his devices and considering his whole history. GUP is represented by a pair (IR, GC), where (i) IR is a list of pair (c, IR_i) such that c is a concept and IR_i is the interest rate computed by the client agent CA_i for this concept, while (ii) GC contains a set of global coefficients relative to each concept visited by the user U. More in particular, each concept c in GC is described by a tuple of the form $\langle c, GIR\rangle$, where:

- c identifies the concept;
- GIR is the *Global Interest Rate* of the concept c, that measures the interest the user U has shown relatively to c by using all his devices. GIR is computed as a weighted mean of all the interest rates IR_i for the concept c stored in the list IR and corresponding to each client agent CA_i. The weight used for each IR_i is computed by the function f, called by passing it as input parameters the i-th row of the matrix DP. The user can choose its personalized function f, as well as the parameters to use for the matrix M. That is:

$$GIR = \frac{\sum_{i=1}^{n} f(DP_{i,1}, DP_{i,2}, .., DP_{i,m}) \cdot IR_i}{\sum_{i=1}^{n} f(DP_{i,1}, DP_{i,2}, .., DP_{i,m})}$$

For instance, in the experiments of Section 4 we have decided to use only a parameter for weighting the contribution of the interest rate coming from each client agent; in this case, the matrix DP becomes a vector $[DP_1, DP_2, .., DP_n]$. We have also chosen to use as DP_i the *price per Mega Byte* estimated for the device i. Finally, we have chosen as function f the linear function. Thus, the formula for computing GIR in this case becomes:

$$GIR = \frac{\sum_{i=1}^{n} DP_i \cdot IR_i}{\sum_{i=1}^{n} DP_i}$$

However, with the general formula for GIR it is possible to use more than one parameter for weighting the contribution of the interest rate of each client agent, and it is also possible to choose any functional form for f.

Server Behaviour. The behavior of SA consists of the following two sub-behaviours:

Setup - In this phase SA is logged into the Agency, after an initial setting of its parameters contained in the Server Profile SP, i.e. the number n of the client agents, the number m of parameters necessary for computing the global interest rates and the choice of a suitable function ψ.

Support - The SA activity consists in: (i) Updating the Global User Profile GUP exploiting the data that each U's client agent periodically sends to SA. This data, for each concept visited by U using the device i, is stored in a tuple of the form $\langle c, IR_i \rangle$; if this visited concept c occurs also in the Global User Profile GUP, the IR_i is stored in the list IR and is immediately exploited to update the global coefficient GIR associated with c; elsewhere, if c is a new concept, visited by the user for the first time, a new element is added in both the list IR and the set of the global coefficients GC (ii) Supporting each U's client agent in the update of its profiles by periodically providing the global coefficient IR for each concept c.

2.5. The Adapter Agent

The third agent type of the MASHA architecture is the *Adapter Agent*. An adapter agent is associated with each monitored Web site in order to support Web activities of both Web site manager and visitors. Each

adapter agent is logged into the *MASHA Agency* and is automatically activated (resp. deactivated) when its Web site is on-line (resp., off-line) or for an explicit web-master's choice.

On the client-side, the user receives a Web site presentation adapted on-the-fly by the adapter agent that shows only the concept instances compatible with the maximum size required by the client. The adapter agent also generates some useful suggestions for the future navigation of the user. Therefore, on the site-side, the adapter agent operates as a recommender system. Firstly, it acts as a content-based recommender system, presenting to the user U in the site visualization only those concept instances which are of most interest to the user, in a way that also takes into account the parameters of the device exploited by U. As a second capability, the adapter agent operates a comparison between the concepts selected by the current visitor and those of the past visitors. In order to perform this comparison, the profile P_i^U of the user U, that is visiting the site using his client agent CA_i, is compared with each profile P_i^O associated to each other user O that has visited the site in the past, by using the same device i. All the profiles P_i^O of these past visitors are stored in a profile collector contained in the adapter agent. The data structure of an adapter agent and its behavior is briefly described below. Without loss of generality we consider an adapter agent AA_W associated to a Web site W.

Adapter Data Structure. The data structure of AA_W is described by the pair (AP_W, PC_W) where:

the **Adapter Profile** AP_W contains three elements, namely (*i*) the AID, that is the *Adapter Agent* identifier; (*ii*) the *Site Catalogue* SC, that is a list containing all the concepts present on the Web pages of the site; (*iii*) the integer z, that represents the number of similar visitors that the adapter selects in order to generate information filtering suggestions for a user;

the **Profile Collector** PC_W contains some information relative to all the client agents that have visited the site in the past. More in detail, $PC_W = [VP_1, VP_2, ..., VP_n]$ is the list of the n *visitor profiles*. Each visitor profile $VP_k = (CP_i, L)$ contains both the client profile CP_i of the visitor k, that exploited the device i, and a concept list L whose elements are pairs (c, IR), where c is a concept and IR is the associated interest rate. These pairs are relative to those concepts that the user k considers the most interesting of the site. Note that we denote by VP_k the profile of a visitor k *that uses a determined device*, and not a global profile of a user. For example, if the same user k visits the site W in two different times by using two different devices, the adapter will store in the

list PC_W two different entries for these two accesses of k, each one relative to a different device. Both the information CP_i and L of each visitor profile VP_k are provided by the client agent CA_i of the user k when it accesses to the site and begins its behaviour (see Section 2.3).

Adapter Behaviour. The behavior of the adapter agent AA_W consists of three sub-behaviours, namely:

Setup - In this phase, AA_W is logged into the Agency that assigns it an AID. Moreover, this behaviour automatically constructs the catalogue of the concepts SC, by simply examining all the Web pages belonging to the site and collecting the current concepts. *Monitor phase* - The adapter agent waits for requests of visit coming from client agents. When a client agent accesses the site W, the adapter sends it the list of all the concepts composing the site. *Support phase* - This phase consists in providing the visitor with a Web site presentation adapted on-the-fly to his characteristics. The support phase is composed of two sub-phases, called *Content-Based Recommendations* and *Information Filtering Recommendations*. These sub-phases perform as follows:

Content-Based Recommendations. When the Web site, monitored by the adapter agent, is visited by a client agent CA_i of a user U that exploits the device i, firstly CA_i sends to the adapter agent its user profile UP_i and then CA_i receives from the adapter agent a list containing those concepts of the site whose concept instances have a size that is compatible with the parameter $MSSet$ contained in the user's profile of CA_i. Then the client agent selects and orders, in a decreasing fashion, those concepts present in the received list that are common with its user's profile and that better meet the user's interests. Such a selection is performed on the basis of the interest degree IR_i of the concepts, that also take into account the exploited device, as previously described in Section 2.3. Finally, the client agent sends to the adapter agent a list L_i containing the k site concepts that have the highest interest rates IR_i. Note that k is a parameter that the user U set in the client profile CP_i. Moreover, we remember that each agent in MASHA community can access to the common shared vocabulary $\mathcal{V}$ containing the list of all the concepts that can be used in Web sites.

Figure 1 shows the details of the content-based recommendation algorithm represented by the function *CBrecommendations* that is executed by the adapter agent each time a client agent CA_i visits

its site. This function accepts as input a client agent CA_i and returns as output the list of concept instances that the adapter will show in the site presentation personalized for CA_i. Firstly, *CBrecommendations* calls the function *extractFromCatalogue* that receives as input the client profile CP of CA_i and returns the list L of concept instances belonging to the site catalogue whose size is compatible with the parameter $MSSet_i$ of CP_i. Then, the function *extractConcepts* is called, that receives as input the list L and returns the list $L1$ of all the concepts of the vocabulary $\mathcal{V}$ that present some instances in L. The list $L1$ is sent to the agent CA_i by using the function *send*. Then, the function *receive* is called, that waits for the response of the client agent. When the response arrives, it is stored in the list of concepts $L2$. Finally, it is exploited the function *findInCatalogue* that accepts as input the list $L2$ and returns the list $L3$ of all the concept instances in the catalogue of the site whose associated concepts belong to $L2$. The list $L3$ is the output returned by *CBrecommendations*. This description shows that the core of the receommendation job is represented by the reception of the list of concepts coming from the client agent CA_i. This list is determined by the client agent as a consequence of the reception of the message sent by the adapter using the function *send*, as shown above. When this message is received, the client agent CA_i executes the function *conceptsOfInterest*, that receives as input a list $L1$ of concept instances and yields as output a list L_i of concepts. First, *conceptsOfInterest* exploits the function *intersection* to construct the list $L2$ which contains the elements that belong to both $L1$ and the list of the concepts of the user's profile UP_i contained in CA_i; this list is constructed by the function *conceptsOf* that receives as input the user profile UP_i. The list $L2$ is then sorted, on the basis of the interest rate, by the function *sort* and finally the first k elements of $L2$ are stored into the list L_i that is produced as output of *conceptsOfInterest*.

Collaborative Filtering Recommendations. The adapter selects, in the set of the n visitors contained in the Profile Collector, the z visitors (where z is a parameter internal to the Adapter, that the Adapter administrator can arbitrarily set) that have the most similar profiles with respect to that of U. The similarity between the user's profile of U and that of another agent x that belongs to the profile collector is computed as follows: Let l be the number of concepts common to both the concept list sent to the adapter by U and x; for each of these concepts, say c_j, $j = 1, .., l$, let d_j be the difference, in absolute value, between the interest rates given

```
ListOfConceptsInstances CBrecommendations(clientAgent CA_i) {
                        ListOfConceptInstances L=extractFromCatalogue(CA_i.CP);
                        ListOfConcepts L1=extractConcept(L);
                        send(L1,CA_i.CP.AID);
                        ListOfConcepts L2=receive();
                        ListOfConceptInstances L3=findInCatalogue(L2);
                        return L3;
}

ListOfConcepts conceptsOfInterest(ListofConceptInstances L1) {
               ListOfConcepts L2=intersection(L1, conceptsOf(UP_i));
               ListOfConcepts L2=sort(L2);\\ sorting on the basis of IR_i
               for(i = 0; i < CP_i.k; i++)
                   insert(L_i,L2[k]);
               return L_i;
}
```

Figure 1. The Content-Based Recommendation Algorithm

to the concept by U and x, respectively. The similarity between U and x is given by the sum of all the contributions $(1 - d_j)$. In such a way, we have considered, when evaluating the similarity, all the concepts that are common to the profiles of x and U. The contribution of each of this concept is maximum (i.e. equal to 1) if U and x give the same interest rate to the associated concept, while it is minimum (i.e. equal to 0) if U (resp. x) gives an interest rate equal to 1 (resp. 0) and x gives an interest rate equal to 0 (resp. 1). For each of the z selected agents, say x, the Adapter shows in the site presentation the concepts instances whose concepts belong to the list L contained in the visitor profile of x. The details of the Collaborative filtering algorithm are reported in Figure 2. The algorithm has been implemented by the function *CFrecommendations* that receives as input the list of concepts L sent by the client agent CA_i by means of the function *conceptsOfInterest* described above and the client profile CP_i. The function *CFrecommendations* returns as output a list of concept instances. This function first constructs a *list of visitors* PL, whose elements are integer values representing the positions, in the Profile Collector of the Adapter, of the z visitors whose profiles have the highest similarities with L. The list PL is obtained as output of the function *mostSimilarVisitors* that receives as input the list L, the client profile CP, the profile collector PC of the site and the integer z. The concept instances belonging to the profiles that correspond to the most similar visitors are stored in the list of concept instances CF by using the function *insert*, and finally CF is returned as output of *CFrecommendations*. Figure 2 shows also the code of the

function *mostSimilarVisitors*. This function first constructs a list LP of pairs $(index, similarity)$ by using the function *similarity* that receives as input the list L and the i-th element $VP[i]$ of the profile collector and returns the pair (i, s_i) where s_i is the similarity between L and $VP[i]$. Note that the function *similarity* is called only for those agents i using the same device as the client agent CA. Then the list LP is ordered on the basis of the similarity values by using the function *order*. Finally, the indices of the first z elements of LP are stored in the list of integers LV that is returned as output of the function *CFrecommendations*.

```
ListOfConceptsInstances CFrecommendations(ListOfConcepts L, ClientProfile CP_i) {
                        ListOfConceptsInstances CF;
                        ListOfVisitors PL=mostSimilarVisitors(L,CP_i,PC,z);
                        for(i = 0; i < z; i++)
                            insert(CF, VP[PL[i]]);
                        return CF;
}

ListOfVisitors mostSimilarVisitors(ListofConceptInstances L, ClientProfile CP,
ProfileCollector PC, int z) {
                ListofVisitors LV;
                \\ the list LP contains pairs (index, similarity)
                ListofPairs LP;
                for(i = 0; i < n; i++)
                    if(CP.device == PC.VP[i].device)
                      LP[i]=similarity(L,PC.VP[i].L);
                order(LP);
                for(i = 0; i < z; i++)
                    insert(LV, LP[i].index);
                return LV;
}
```

Figure 2. The Collaborative Filtering Recommendation Algorithm

The proposed collaborative filtering technique is a variant of the traditional collaborative filtering to generate recommendations (Breese et al., 1998), and similarly to the original technique it is computationally expensive. If n is the number of site visitors and m is the number of concepts in the site catalogue, the computational complexity of the technique is $\mathcal{O}(m \cdot n)$ in the worst case, since it examines n visitors and up to m concepts for each visitor. However, because the average visitor profile is extremely sparse, the algorithms performance tends to be closer to $\mathcal{O}(m+n)$. If we have a huge number of agents and a huge size of site catalogue, the scalability of the algorithm becomes an issue to face. To address such an issue, a variant to the traditional collaborative filtering has been proposed called *item-to-item* collaborative filtering (Linden et al., 2003). This technique, rather than matching the current

agent to similar visitors, compares each of the accessed concepts to similar concepts, then combines those similar concepts into a recommendation list. In order to determine the most similar match for a given concept, a similar-concepts table it is exploited by finding concepts that visitors tend to access together. Given a similar-concepts table, the item-to-item technique finds concepts similar to each of the current visitor's profile, aggregates those concepts, and then recommends the most interesting among those concepts. This computation is very efficient, because it only depends on the number of concepts the current visitor accessed. The similarity of two concepts, $C1$ and $C2$, can be measured in various ways; a common method is to measure the cosine of the angle between the two vectors $V1$ and $V2$ associated to $C1$ and $C2$, where $V1$ (resp. $V2$) contains the interest rate of those visitors that accessed $C1$ (resp. $C2$). We have implemented in MASHA, as an alternative to the traditional collaborative filtering, also an extension of the item-to-item technique described above. This extension considers, in computing the similarity between two concepts, two vectors $V1$ and $V2$ containing interest rates relative to visitors exploiting the same device.

3. Related Work

An overwhelming amount of Recommender Systems based on the use of software agents has been proposed in the last years (Montaner et al., 2003). Accordingly with the usual classification of the recommender systems based on the exploited recommendation method (Adomavicious and Tuzhilin, 2005; Burke, 2002), we identify the following categories: (A) *Content-based*: the user will be recommended with items similar to the ones preferred by him in the past; (B) *Collaborative*: the user will be recommended with items that people with similar tastes and preferences have selected in the past; (C) *Hybrid approaches*: these methods combine collaborative and content-based methods. In this classification MASHA belongs to the last category.

On the other side, an emerging issue is represented by the adaptivity (Riecken, 2000) and particularly by the device adaptivity (Al-Bar and Wakeman, 2001) in order to take into account both user's preferences and device capabilities. A lot of approaches have been proposed in the literature but, to the best of our knowledge, none of them considers devices capabilities in the users' profiles construction. Consequently, in these systems recommendation and adaptivity phases are assumed to be orthogonal.

In this section will be presented both recommender systems that do not implement device adaptivity and systems taking into account device adaptivity issues. Finally, differences and similarities with MASHA will be highlighted.

3.1. Recommender Systems

SUGGEST (Silvestri et al., 2004) supports user navigation on the Web by dynamically generating links to pages that have not been visited in the past by a user and might be potentially interesting to him. It is also capable of managing dynamic Web sites. In order to carry out its task, SUGGEST builds and maintains historical information about user behavior by means of an incremental graph partitioning algorithm. In order to extract information about navigational patterns, SUGGEST exploits an algorithm that models the usage information as a complete graph $G = (V.E)$. The set V of vertices contains the identifiers of the different pages hosted on the Web server. The set of edges E is weighted by using the following relation: $W_{ij} = N_{ij}/max\{N_i, N_j\}$, where N_{ij} is the number of sessions containing both pages i and j, N_i and N_j are respectively the number of sessions containing only page i or page j. The users' sessions are identified by means of cookies stored on the client side. The so constructed graph is then partitioned by using a clustering algorithm, in order to find groups of strongly correlated pages. The algorithm is a modified version of the well-known incremental connected components algorithm. After the clustering step, a suggestion list is constructed in a straightforward manner, by finding the cluster which has the largest intersection with the page window correspondent to the current session.

COMTELLA (COMTELLA, 2006) (by COMmunity GnuTELLA) has been developed at the MADMUC Lab at the University of Saskatchewan and it is a Gnutella-based P2P application that allows a users group to share resources (i.e., services or files). COMTELLA provides incentives to encourage the users participation and cooperation. The users are supposed to rate the resources provided by other participants; such rates are then aggregated into recommendation lists, that implement in such a way a social-based recommender. This mechanism is easily usable in multi agent frameworks (as in (Wang and Vassileva, 2004) to form P2P communities to create automated kinds of user support.

The **C-Graph** (Buccafurri et al., 2002) approach proposes an agent model, able to support the navigation of a Web user, providing him with useful suggestions. The C-Graph agent, during the navigation of a site, carries out, at the same time, two activities: the first is providing the user with a set of recommendations for supporting his navigation,

and the second is monitoring the user behavior in order to learn his preferences and encoding them into a user profile as a new piece of knowledge. A key issue for this system concerns the model used to represent the user profile which is capable of embedding both concepts of interest to him and the correlations he perceives. This user profile is a rooted labelled direct graph $\langle N, A\rangle$, where N is the set of nodes and $A \subseteq N \times N$ is the set of arcs. Informally, N represents the set of concepts of interest for the user u. The arcs encode semantic relationships among concepts. Their labels define a number of properties associated to relationships also containing the dependency of the model on the user u. More precisely, an arc (s,t) is provided with a label $l(s,t) = \langle d_{st}, r_{st}, h_{st}, \tau_{st}\rangle$, where both d_{st} and r_{st} are real numbers ranging from 0 to 1, h_{st} is a non negative integer, and τ_{st} is a non negative real number. The four *label coefficients* above encode different properties, that can be computed analyzing the visited documents and expressing some kind of relationships among concepts. The approach defines two synthetic functions, called ψ and ρ, the former encoding the structural closeness, the latter the user preference. In order to have a unique synthetic information summarizing both structural and behavioral components a function γ is defined, measuring the "subjective" semantic closeness of two concepts. The user can set, in computing the semantic closeness, the degree of importance he gives to the structural preference with respect to the behavioral one, by setting an internal parameter k. More precisely, the semantic closeness between two concepts s and t is $\gamma(s,t) = k \cdot \psi(s,t) + (1-k) \cdot \rho(s,t)$.

X-Compass (Garruzzo et al., 2002**)** is an XML-based agent model for supporting a user during his activities on the Web. For this purpose, X-Compass constructs and manages a user profile; such an activity is performed automatically, by monitoring the behaviour of a user during his accesses to Web pages. This profile represents the user's interests, as well as some relationships existing among them. Two different kinds of relationships are handled in the user's profile, namely: *(i) is-a relationships*, organizing user interests in a generalization hierarchy; *(ii) associative relationships*, linking user's interests appearing distant in the is-a hierarchy but determined to be close according to the analysis of the user's behaviour; the extraction of these relationships requires the exploitation of data mining techniques. After the user profile has been constructed, X-Compass exploits it to perform both content-based and collaborative filtering recommendation activities. In particular, each user u is supported by an agent $Ag(u)$, that stores the following data structures; (i) A *user profile* $P(u)$, storing interests and preferences of u. It is a rooted graph, where each node i represents a concept of interest for u. Two pieces of information are associated with i, namely an

attraction degree $DAttr_i$, representing the degree of interest for i, and a key set $KSet_i$, defining the semantics of the interest associated with N_i. The arcs of $P(u)$ represent both is-a relationships and associative rules; (ii) A *history* $H(u)$ represents an ordered list of elements, each associated with an access to a Web page performed by u; the order of $H(u)$ reflects the temporal succession of accesses; an *aggregated history* $AH(u)$, that is a list of elements, each associated with a Web page. Each element represents the whole past history of the user when visiting the associated Web page. During each session, the agent monitors each access to a Web page carried out by u and extracts the necessary information; then, it updates H, AH and the Attractiveness Degree of the Representative Node of the currently visited page. After this, it exploits P for providing content-based recommendations (as an example, for suggesting to u the next page to visit). At the end of each session, P, H and AH are updated; the new P is, then, exploited for providing u with content-based or collaborative filtering recommendations.

DESIRE (Parsons et al., 2004) (Desirability Estimator and Structured Information Recommendation Engine) is a content-based Recommender System that combines a viewing-time and attribute-based preference inference algorithm with an attribute-based recommendation engine. In order to make its recommendations, DESIRE exploits only the current user navigational data, in conjunction with item property data.

Duine (Van Setten, 2005) is a prediction framework adopting a hybrid approach. More precisely, it implements a switching method to choose the prediction technique, among those available, able to provide more accurate recommendations. Unlike other hybrid recommender systems developed for specific domain, Duine is domain-independent. Tests referred to different applications (a TV program guide and a movie recommender system) demonstrate in these cases the effectiveness of the multi-prediction approach adopted by Duine with respect to a single-prediction strategy.

Community Search Assistant (Glance, 2001) is a system that has been conceived for supporting users of search engines to find information related to what they are currently accessing. In order to carry out its task, Community Search Assistant allows the users in a community to realize collaborative searches. The system exploits a support graph whose nodes represent queries submitted by the community users and whose arcs denote links between the queries. In this way, the single user exploitation of the information network is transformed into a collaborative usage of it, since users can tap into the knowledge base of queries submitted by others.

METIOREW (Bueno et al., 2001) is an "objective-oriented" system for supporting Web navigation; in the context of METIOREW, an objective represents an information need. Initially a user inserts his objectives by providing a list of keywords for each of them and the system constructs his profile as the set of his objectives. After this, METIOREW associates two models with the new user, namely his own model and the most similar one already present in its knowledge base; this latter is associated with the new user until his own model becomes significant. METIOREW provides its recommendations to a user on the basis of both his profile and the support one (if the user is new).

3.2. Adaptive Systems

INTRIGUE (Ardissono et al, 2003) joins the recommender and the adaptive hypermedia issues in a multi agent prototype to support tourists' activities using a XML-based approach. INTRIGUE generates appropriated sightseeing tours for heterogeneous tourist groups taking into account preferences (individually or for homogeneous subgroups) and environmental constraints, which should be preliminary declared to the system. Each personalized recommendation is represented by means of an XML document, independently by the device. This XML document is analyzed, exploiting the XML meta-information, to create, with XSLT (eXtensible Stylesheet Language Trasformation) transformations, a HTML/WAP page, able to satisfy both browser and device constraints.

Proteus (Anderson et al., 2001) is a Web site personalizer that acts as an intermediary between a Web site and its visitors adapting automatically the Web contents to the mobile devices of the visitors in order to save users' time and efforts during their visits to the site. Proteus adapts the site content transforming the site organization based on the expected utility of the visitor. As a result, the pages considered more interesting for the user will be highlighted, while the remaining site content will still remain accessible. To accomplish its task Proteus extracts from the site its contents and links and exploits visitors' information such as their navigation actions (scrolls down the page, follows a link, etc.), demographic class and so on.

AHA! (De Bra et al., 2002; AHA! project, 2006). The Adaptive Hypermedia Architecture (AHA!) is an Open Source Web-based adaptation engine written in Java and originally developed to support an on-line course. AHA! adopts a multi-level architecture based on some server-side software called models. These models, described either as XML or in mySQL table, specify how a Web page required by a user must be automatically adapted to the user's preferences in terms of

presentation and navigation support. AHA! reaches its goals in more steps by which, for each Web page required by the user, the correlation with other Web pages is extracted and adapted on the basis of the user's preferences and modalities he/she usually adopts to correlate concepts and pages. Furthermore, the user could adopt different modules to take into account different devices and consequently different adaptivity criteria, but the adaptive processes will be separate for each device issue.

3.3. Similarities and Differences with MASHA

All the aforementioned systems exploit, similarly to MASHA, an internal profile for storing information relative to the user.

The main differences with MASHA are the following: (*i*) each of the described systems stores a user profile in a unique client agent, able to automatically construct and maintain the profile; differently, MASHA stores an analogous profile into a server agent that constructs it on the basis of the information coming from the different user's client agents. This leads to the advantage that the MASHA client agent only collects information about the associated user, while the actual profile construction is left to the server, making the task of the single client lighter; (*ii*) none of the above described systems considers the user's devices in the construction of a unique user's profile to be exploited in a recommendation stage; (*iii*) none of these recommender system takes into account the exploited device in the recommendation algorithm, while MASHA uses this information in the collaborative filtering and also its content-based recommendations are influenced by the consideration of the device in the profile construction.

Moreover, concerning the systems that carry out the adaptivity, we point out that: (*i*) INTRIGUE, Proteus and AHA! do not exploit any information about the device to construct their user profiles, unlike MASHA; (*ii*) Proteus adopts a client/server multi agent framework like MASHA does, but it exploits a complex structure to extract user preferences and site-content and this absorbs great computational resources, while MASHA is a realtime system; (*iii*) MASHA and INTRIGUE are similar in the Web content description, since both exploit XML; (*iv*) INTRIGUE and Proteus do not take into account the device in their recommendation stages unlike MASHA; (*v*) AHA! carries out the adaptivity in more steps adopting a multi-level process; in this way this process is more sophisticated and versatile than MASHA but it is not adaptable to each device requiring different user profile modules;

4. Experiments

In this Section, we present some experiments devoted to evaluating the capability of our multi agent system to support both content-based, collaborative filtering-based and device-based adaptivity. To this purpose, we have compared MASHA performances in generating suggestions with those of other three recommender systems, namely SUGGEST, X-COMPASS and C-GRAPH. We have chosen X-COMPASS and C-GRAPH since (*i*) they are, like MASHA, both content-based and collaborative filtering and they exploit a user profile; (*ii*) they are, at the best of our knowledge, two of the most performative recommender systems. Moreover, since we want to analyze separately the contribution of the content-based and the collaborative filtering algorithms, we have chosen also the content-based system SUGGEST that turns out to be one of the best performing in this context. In the following sub-sections we describe the Web sites and the software programs that we have created for carrying out our experiments.

4.1. Web Sites Population

We have built 30 different Web sites related to Travel Agencies. Accordingly with the assumption made in Section 2.1, the pages of each site are created as XML documents, by using a common vocabulary represented by a unique XML Schema; therefore each page contains only instances of this XML schema. We denote the whole population of Web sites as WP. As an example, one of these Web sites is shown in Figure 3. We can see that the site describes some available holiday destinations, and also contains a section *the other users suggest*, presenting the suggestions given by the site users. The whole collection of the test sites used in this experiment are available on the MASHA project site (MASHA, 2006).

4.2. User Population

We have monitored the actions of 104 real users, belonging to a user population U, that navigated in the Web sites belonging to WP, without using any recommendation support. Each of these users has exploited three different devices, namely a desktop PC, a palmtop and a cellular phone. The choices made by the users have been recorded in 104 log files, one for each user, that are available at the MASHA project site (MASHA, 2006). Each log file, associated to a user u, contains a list of 500 elements $\langle s, d, t \rangle$, related to 500 different clicks performed by the user in a period of 20 days, where s (resp. d) is the identifier of

the *source* (resp. *destination*) concept instance, and t is the time of the choice, made by u, to go from s to d via a hyperlink.

4.3. Agent Population

We have built three types of software agents, namely *client*, *server* and *adapter*. All these agents have been developed by using JADE (Java Agent Development Framework), as extensions of the basic class *Agent*. In order to realize the clients for palmtop and mobile phones, we have exploited JADE/LEAP (JADE Lightweight Extensible Agent Platworm), a reduced version of JADE operating on devices with limited resources. We have chosen JADE since (*i*) it offers a lot of built-in functions for efficiently managing agent communication; (*ii*) it can choose the proper message transportation mechanism by considering the agent location, thus lightening the programmer's work because he has no need to know the physical addresses of agents; (*iii*) it provides specific primitives for handling messages passing in wireless environments; (iv) it offers good scalability. Some of these agents are MASHA agents (namely client, server and adapter agents) that implement our approach of generating user suggestions. The other agent types are client agents built by following the recommendation-based approaches called SUGGEST, C-GRAPH and X-COMPASS, that we have presented in Section 3 and that we compare in this Section with our approach.

Client Agents. We have built, for each user belonging to U, the following client agents:

MASHA Client Agents. We have three client agents associated with three different devices, namely a PC, a palmtop and a cellular phone. We have used the following parameter for the client agent associated to the PC: $T_M = 200$ sec (i.e., we have assumed that the interest for a page visited by a PC is "saturated" after 3 minutes and 20 seconds); $\rho_1 = 0.6$, $\rho_2 = 0.8$, $\rho_3 = 0.9$, that is we have considered that a user that exploits a PC and stores the page containing a concept instance has an interest rate higher of about 20 percent with respect to the case of the user who simply visits the page. Moreover, we have considered the user that prints the page to be 10 percent more interested than the user storing the page. Instead, we have used the following parameters for the client agent associated to the palmtop: $T_M = 120$ sec (i.e., this means to assume the interest for a page visited by a palmtop as "saturated" after only 2 minutes, with a reduction of 40 percent with respect to the case of the PC); $\rho_1 = 0.6$, $\rho_2 = 0.9$, $\rho_3 = 1$, by thus increasing the difference of interest between the case of

a simple visit and those of either storing or printing. Finally, we have set the following parameters for the client agent associated to a cellular phone: $T_M = 60$ sec (i.e., we have considered the interest for a page visited by a cellular as "saturated" after only 1 minute, with a reduction of 50 percent with respect to the case of the palmtop); $\rho_1 = 0.5$, $\rho_2 = 0.9$, $\rho_3 = 1$, by further increasing the difference of interest between the case of a simple visit and those of storing or printing. Moreover, we have set the parameters P and ψ of all the client agents associated to a PC equal to 3 and 0.9, respectively, meaning that we have chosen to decrease of 10 percent the interest for a concept that is not accessed for 2 days; Instead, the same parameters P and ψ of all the client agents associated to either a palmtop or a cellular phone are equal to 3 and 0.95, respectively; this means that we have decided to perform a less intensive decreasing of the interest for these typologies of devices (only 5 percent for three consecutive days of no access for the concept), due to the probable less frequent use of them. Finally, all the clients have the coefficients k equal to 4, meaning that the user desires that the adapter shows him the four most interesting concept instances.

Other Client Agents. We have also constructed, for each user belonging to U, a SUGGEST client agent, a C-GRAPH client agent and an X-COMPASS client agent. These agents are built by following the descriptions of the relative data structures and recommendation algorithms proposed in (Silvestri et al., 2004; Buccafurri et al., 2002; Garruzzo et al., 2002), respectively. More particularly, relative to the C-GRAPH agent, we have used a coefficient $k = 0.5$ for giving the same importance to both the structural and semantic closeness.

Server Agents. Moreover, each user of U is associated with a MASHA server agent. The values of the parameters of each server agent are the same for all the users: $n = 3$, since we have three types of client agents for each server agent; $m = 1$, since we have decided to use only a parameter for weighting the contribution of the interest rate coming from each client agent; in this case, the matrix DP becomes a vector $[DP_1, DP_2, DP_3]$. We have chosen to use for DP_1 (resp. DP_2,DP_3) the *price per Mega Byte* estimated for a PC (resp. palmtop, cellular phone). The prices per Mbyte (in euro cents) that we have considered are: $DP_1 = 0.9$,$DP_1 = 1.4$, $DP_3 = 1.8$, thus the formula for computing the global interest rate GIR is: $GIR = \frac{\sum_{i=1}^{3} DP_i \cdot IR_i}{\sum_{i=1}^{3} DP_i}$

Site Agents. We have provided each site of W with a MASHA adapter agent. On each site, the adapter agent can be set in a *default mode*, meaning that the recommendations are disabled. In this case the site presentation will be standard and static, not depending on either the profile or the device of the user that visits the site. As an example, Figure 3 shows the standard presentation of a Web site of WP. If the recommendations are enabled, the adapter generates different presentations by using its recommendation algorithm.

4.4. Description of the experiments

In our experiments, each user of U visits some Web sites belonging to W. Each visit is relative to a concept instance c, and if the user, at the time $t(x)$, goes from a concept instance x to another concept instance by following a hyperlink, we denote this latter concept instance as $next(x)$. We have collected, for each user, the first 750 triplet $(x, next(x), t(x))$ relative to his navigation. The first 400 triplet are used as a training-set for allowing the agents of the visitor to construct their user profiles. The remaining 350 triplet are used as a test-set for evaluating the recommendation algorithm used by the site agents. That is, for each user, in correspondence of each triplet $(x, next(x), t(x))$ belonging to the test-set, we have generated the recommendations $R_i(x)$, $i = 1, 2, 3, 4$ of the four algorithms MASHA, X-COMPASS, C-GRAPH and SUGGEST, respectively. Each recommendation $R_i(x)$ is a list of recommended concept instances. We compare x and $R(x)$ in order to measure the effectiveness of the different approaches. We have used, as a measure of effectiveness, the performance metrics, precision, recall and F-measure, accordingly with most of the related work (Kim et al., 2004). Precision is defined as the share of the concepts actually visited by u among those recommended by the system; vice versa, Recall is the share of the concepts suggested by the system among those chosen by u. F-Measure represents the harmonic mean between Precision and Recall. Precision, recall, and F-measure can be represented as follows.

$$Pre(R(x)) = \frac{|R(x) \bigcap next(x)|}{|R(x)|}$$

$$Rec(R(x)) = \frac{|R(x) \bigcap next(x)|}{|R(x)|}$$

$$F(R(x)) = \frac{2 * Rec(R(x))) * Pre(R(x)))}{Rec(R(x)) + Pre(R(x)))}$$

Table I shows the values of the Average Precision, the Average Recall and the Average F-Measure obtained, in this experiment, by the four considered approaches.

Table I. Performances of four different recommendation algorithms

	MASHA	*X-COMPASS*	*C-GRAPH*	*SUGGEST*
Pre	0.312	0.214	0.233	0.170
Rec	0.184	0.101	0.087	0.079
F	0.241	0.151	0.148	0.130

Table II. The effect of using a unique device on the comparison results

	MASHA	*X-COMPASS*	*C-GRAPH*	*SUGGEST*
Pre	0.210	0.214	0.233	0.190
Rec	0.094	0.101	0.087	0.089
F	0.149	0.151	0.148	0.142

This table shows that the values of the accuracy measures relative to MASHA are about 25 percent higher than that of C-Graph (the best of the other approaches chosen for the comparison, in terms of accuracy). MASHA is also the approach having the highest recall, performing about 45 percent better than X-Compass (the best of the other approaches chosen for the comparison, in terms of recall). This very good performance is synthetically evaluated by the F-measure, that for MASHA is about 37 percent higher than that of the other approaches, that have similar values. These results show that MASHA obtains better results, as a recommender system, than the other considered approaches. We argue that this is due to the fact that MASHA agents consider, in determining its suggestions, also the devices that have been exploited by the user. For evaluating the effect of considering device influence, we have repeated the above mentioned comparisons, by using a unique MASHA client agent (the one we used in the previous experiment in association with the device PC), instead of three different clients. In this way, the MASHA approach generates recommendations that do not take into account the effect of the different devices exploited by the user in the past. Results of this experiment are shown in Table II.

As we can see, MASHA approach shows, in these conditions, performances comparable with, but no higher than those of the other approaches, and this confirms that the main advantage of our approach consists in the introduction of different client agents for the same user, associated to the different devices that he exploits. In order to further clarify this point, we have repeated this experiment by using as a unique device the palmtop and the cellular phone, respectively. The result obtained has been the same: MASHA performs similarly to the other systems if a unique device is exploited, while it performs significantly better in presence of multiple devices. In order to understand more precisely how such a device consideration improves the recommendation performances, we have repeated the experiment of generating suggestions by considering separately the content-based component and the collaborative filtering component of the experiments. That is, we have generated the recommendations of the four systems MASHA, X-COMPASS, C-GRAPH and SUGGEST only taking into account the concepts deriving from the similarity between the profile of the visitor and the content of the site, without considering the concepts suggested by the other users. Since SUGGEST is only a content-based recommender, the suggestions so generated are in this case the same as those of the previous experiments. The results of this experiment are reported in Table III and show that the performances of MASHA is about 16 percent higher, in the F-measure, than those of the other systems. This confirms the supposition that constructing a user profile that takes into account the device exploited in accessing the concepts leads to model more precisely the user preferences, and this positively influences the suggestion performances. Furthermore, we have repeated the experiments considering, for only the three approaches that act also as collaborative filtering methods, those concepts deriving from the suggestions of the other users, without taking into account the content-based component. The result of this experiment is reported in Table IV and shows that the performances of MASHA are in this case significantly improved (about 40 percent higher than the other two approaches in the F-measure). We argue that this is the effect of having considered, in generating collaborative filtering recommendations for a user u, only those users that exploited the same device of u.

Finally, we have analyzed the effect of using, as collaborative filtering method, the item-to-item technique described in Section 2.5. Table V shows precision, recall and F-measure for MASHA using the traditional collaborative filtering (MASHA-T) and the item-to-item collaborative filtering (MASHA-I). The last column of the table also reports the average number N of operations that has been necessary for generating collaborative filtering recommendations, computed on the whole set of

Table III. Content-based component of the performances

	MASHA	*X-COMPASS*	*C-GRAPH*	*SUGGEST*
Pre	0.193	0.165	0.172	0.180
Rec	0.099	0.076	0.074	0.079
F	0.151	0.123	0.125	0.130

Table IV. Collaborative filtering component of the performances

	MASHA	*X-COMPASS*	*C-GRAPH*
Pre	0.141	0.087	0.082
Rec	0.096	0.052	0.047
F	0.119	0.071	0.065

the users. We note that MASHA-T and MASHA-I show very similar quality in the recommendations, but the efficiency of MASHA-I is about 22 percent higher.

The benefits derived from the consideration of the exploited device in generating recommendations are not limited to the production of good quality suggestions. The MASHA adapter uses the information deriving from the client profile of the visitor for adapting the graphical aspect

Table V. Comparison between Traditional and Item-to-Item Collaborative filtering

	MASHA-T	*MASHA-I*
Pre	0.141	0.0137
Rec	0.096	0.094
F	0.119	0.0116
N	121	98

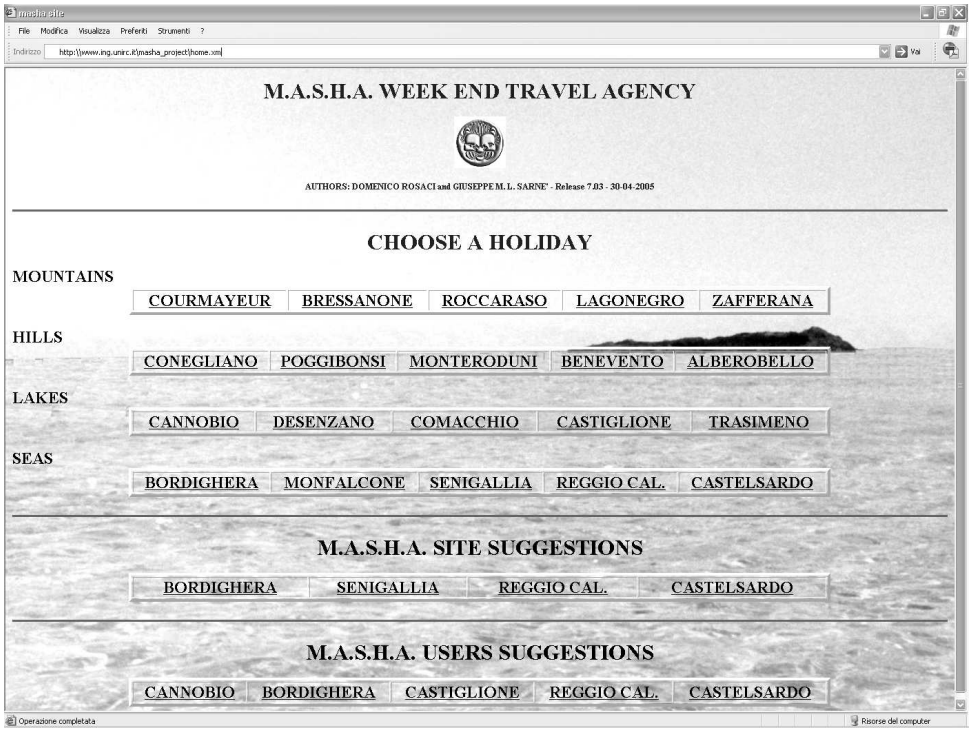

Figure 3. The standard presentation of a MASHA Web travel agency.

of the presentation. As an example, Figure 4 shows the presentation chosen by the Adapter of the same site related to the example of Figure 3, for a visitor that accesses by a palmtop. Note that the graphical presentation is in this case lighter than that related to a PC access, and the suggested destinations are only a subset of all those available, generated following the preferences of the user. As another example, Figure 5 shows the presentation chosen for a visitor that uses a cellular phone. In this case the presentation is merely textual and the adapter suggests only the three destinations that best match with the user's preferences.

5. Conclusions

In this paper we have presented a multi agent system architecture, called *Multi Agent System Handling Adaptivity* (MASHA), designed for handling Web Site adaptivity with respect to both user profile and exploited device. More specifically, our system aims at supporting both users in performing their Web activities according to their past behav-

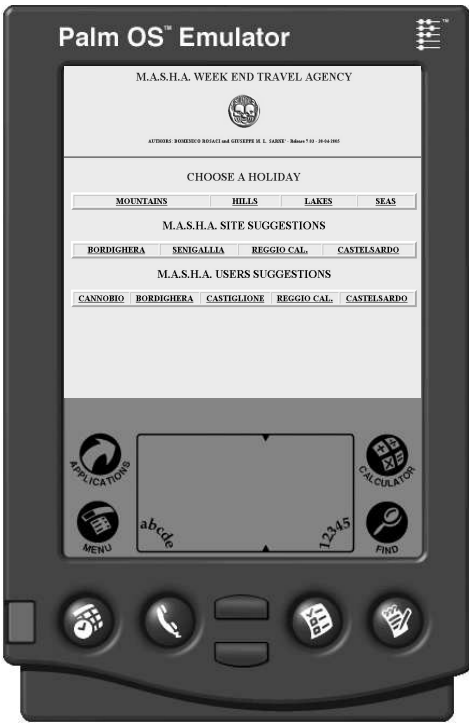

Figure 4. The palmtop presentation of a MASHA Web travel agency.

ior and Web site managers in selecting the most suitable presentation for each typology of visitors. In particular, on one hand, the system is able to generate a Web site presentation adaptive with respect to the profile of both the customers and their exploited devices; on the other hand, the system provides the visitors of a Web site with useful suggestions, acting as both a content-based and a collaborative-based recommender system and also in this context the exploited device is taken into account. We have performed some experiments for evaluating the performances of our systems, in comparison with three other agent-based recommender systems, and the obtained results show a significative improvement in the quality of the suggestions. We highlight that a current, important limitation of MASHA is represented by the assumption of uniformity in the representation of the concepts. This limitation does not allow the use of MASHA in agent environments having heterogeneous representations, as in the case of most of the actual Web sites. MASHA can be fruitfully exploited in environments where a unique vocabulary of concepts has been defined, as in the case of Ebay. However, our ongoing research is planning to extend MASHA for working also in heterogeneous environments, as in HTML Web

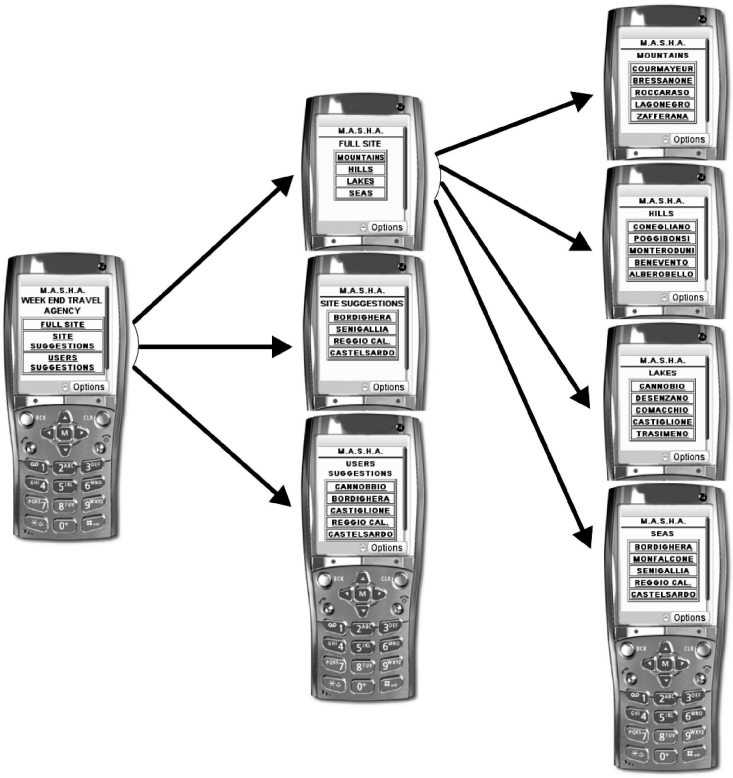

Figure 5. The mobile presentation of a MASHA Web travel agency.

pages. The idea consists in providing the adapter agents of MASHA with the capability of automatically constructing the unique vocabulary, regardless of the markup language (HTML, XML, etc.) exploited for codifying the Web sites. This way, adapter agents would act as wrappers able to extract from the original Web site representation the concepts in the unified representation. Finally, we note that the MASHA system leaves the user free to choose the level of detail in the site presentation, by setting the coefficients k and $MSSet$ of his client profile in a customized way. If the user chooses a high value for both k and $MSSet$, the result will be that all the concept instances of the site will appear in the presentation. We have not dealt in this paper with the issue of partitioning the elements of the presentation in prioritized classes, but we argue that such a partition could be easily carried out by suitably using the interest rate of the concepts. An algorithm for realizing this interesting idea is the object of our ongoing research.

References

G. Adomavicious and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 734-749, 17(6), 2005.

AHA! project URL. http://aha.win.tue.nl. 2006.

A. Al-Bar, and I. Wakeman. A survey of adaptive applications in mbile computing. In *Proc. of the 21st International Conference on Distributed Computing Systems Workshops*, 248–tttt, Meza, AZ, USA, 2001, IEEE.

C. R. Anderson, P. Domingos, and D. S. Weld. Adaptive web navigation for wireless devices. In *Proc. of the 17th IJCAI Conference*, pp. 879–884, Seattle, USA, 2001. Morgan Kaufmann.

L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso. INTRIGUE: Personalized recommendation of tourist actractions for desktop and handset devices. *Applied Artificial Intelligence: Special Issue on Artificial Intelligence for Cultural Heritage and Digital Libreries*, 687-714, 17(8-9), 2003.

L. Ardissono, A. Goy, G. Petrone, M. Segnan, L. Console, L. Lesmo, C. Simone, and P. Torasso. Agent technologies for the development of adaptive web stores. In *Agent Mediated Electronic Commerce, The European AgentLink Perspective*, pp. 194–213. Springer, 2001.

J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *14th International Conference on Uncertainty in Artificial Intelligence*, 43–52. Morgan Kaufmann.

D. Bueno, R. Conejo, and A. David. METIOREW: An objective oriented content based and collaborative recommending system. In *Proc. of Hypermedia: Openess, Structural Awareness, and Adaptivity International Workshops*, pp. 310–314, Aarhus, Denmark, 2001. Springer.

F. Buccafurri, G. Lax, D. Rosaci, and D. Ursino. A user behavior-based agent for improving web usage. In *Proc. of the ODBASE Conf*, pp 1168–1185, Irvine, USA, 2002. Springer.

R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

D. Calvanese, D. De Giacomo and M. Lenzerini. A Framework for Ontology Integration. *The Emerging Semantic Web, Selected Papers from the First Semantic Web Working Symposium*, 201–214, 2002. IOS Press.

P.K. Chan. Constructing Web User Profiles: A non invasive learning approach. *Web Usage Analysis and User Profiling*, 39–55, Springer, 2000.

COMTELLA URL. http://bistrica.usask.ca/madmuc/comtella.htm. 2006.

P. De Bra, A. Aerts, D. Smiths, and N. Stash. AHA! The next generation. In *Proc. of the thirteenth ACM conference on Hypertext and hypermedia*, 21–22, College Park, Maryland, USA, 2002.

FIPA URL. http://www.fipa.org. 2005.

F.J. Garcia, F. Paternò, and A.B. Gil. An adaptive e-commerce system definition. In *Proc. of the Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems*, 505–509, Malaga, Spain, 2002. Springer.

S. Garruzzo, S. Modafferi, D. Rosaci and D. Ursino. X-Compass: an XML agent for supporting user navigation on the Web. In Proc. of 5th FQAS Conf., 97–211, Copenaghen, Denmark, 2002. Springer.

N.S. Glance. Community search assistant. In *Proc. of the Int. Conf. on Intelligent User Interfaces*, 91–96, Santa Fe, New Mexico, USA, 2001.

T.R.. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993.

R.H. Guttman, A.G. Moukas, and P.Maes. Agent-mediated electronic commerce: A survey. *Knowledge Engineering Review*, 13(2):147–159, 1998.

JADE URL. http://www.jade.tilab.com. 2005.

D. Kim, V. Atluri, M. Bieber, N. Adam, and Y. Yesha. A clickstream-based collaborative filtering personalization model: towards a better performance. In *Proc. of the Int. Work. on Web Information and Data Management*, 88–95. ACM, 2004.

R. Lau, A. Hofstede, and P. Bruza. Adaptive profiling agents for electronic commerce. In *Proc. of the CollECTeR Conference on Electronic Commerce*, Breckenridge, USA, 2000.

G. Linden, B. Smith, and J. York. Amazon.com recommendations. item-to-item collaborative filtering. *IEEE Internet Computing*, 76–80, January-February, 2003.

MASHA_Project URL. http://mashaproject.altervista.org. 2006.

M. Montaner, B. López, and J.L. De La Rosa. A taxonomy of recommender agents on Internet. *Artificial Intelligence Review*, 285-330, 19, 2003.

The North America Industry Classifications (NAICS) website, http://www.census.gov/naics. 2005.

J. Parsons, P. Ralph, and K. Gallagher. Using viewing time to infer user preference in recommender systems. In *Proc. of the AAAI Workshop on Semantic Web Personalization*, 52–64, San Jose, USA, 2004. AAI.

P. Resnick, and H.R. Varian. *Communications of the ACM: Special issue on recommender systems*, 40(3), 1997.

D. Riecken. *Communications of the ACM: Special issue on personalization*, 43(8), 2000.

B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proc. of ACM Conference on Electronic Commerce*, pages 158–167, Minneapolis, USA, 2000. ACM.

J.B. Schafer, J.A. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1-2):115–153, 2001.

F. Silvestri, R. Baraglia, P. Palmerini, and M. Serranò. On-line generation of suggestions for web users. *J. Of Digital Information Management*, 392–397, 2(2), 2004.

M. Van Setten. Supportin people in finding information. Hybrid recommender system and goal-based structuring. Telematica Instuut, 2005.

Y. Wang, and J. Vassileva. Trust-based community formation in peer-to-peer file sharing networks. In *Proc. of IEEE/WIC/ACM International Conference on Web Intelligence*, 341–348, Beijing, China, 2004.

Curriculum Vitae

Dr. Domenico Rosaci is Assistant Professor of Computer Science at the University "Mediterranea" of Reggio Calabria. He received his Ph.D. in Computer Science from the University of Reggio Calabria in 1999. Dr. Rosaci has worked in several areas of information systems and artificial intelligence, including knowledge representation, user modeling, OLAP and artificial neural networks. His primary interest lies in the areas of intelligent agents, specifically in multi-agent system modeling. He has authored over sixty scientific papers, published both in international journals and proceedings of international conferences.

Dr. Giuseppe M.L. Sarné is Assistant Professor of Computer Science at the University "Mediterranea" of Reggio Calabria. He works in the fields of user modeling, intelligent agents and multi-agent systems, although previous research has included work in artificial neural networks. He is the author of about forty scientific and technical papers, published both in international journals and proceedings of international conferences.