Draft Version

A complete version of this paper has been published in:
Information and Software Technology, 48, pp. 107-120, 2006.

MAPIS, a multi-agent system for information personalization

Christelle PETIT-ROZÉ, Emmanuelle GRISLIN-LE STRUGEON
LAMIH-UMR CNRS 8530
University of Valenciennes
Le Mont Houy, F-59313 Valenciennes cedex 9, France
Tél: (33) 27511499 - Fax: (33) 27511316
{christelle.roze, emmanuelle.grislin}@univ-valenciennes.fr

Abstract
In the domain of multi-user and agent-oriented information systems, personalized information systems aim to give specific and customized responses to individual user requests. In addition to the ability to analyze user needs and to retrieve, understand and act on distributed data that is offered by any agent-oriented system, multi-agent systems also offer interesting possibilities for interaction, particularly with regard to information sharing and task coordination. Our approach exploits these interactive possibilities in order to make the system capable of personalizing information. In addition, reusable models at both the social and individual levels were chosen for this approach in order to facilitate subsequent information system design. With these two ideas in mind, several models of agent interaction (social) and the internal activity cycles (individual) have been proposed with the aim of creating a multi-agent system for information personalization.

Keywords: Multi-agent systems, interaction model, information personalization

1 Introduction and motivation

Being able to personalize information becomes more important as the volume of the acquired data grows, or becomes more heterogeneous and/or more widely distributed. In a multi-user context, personalized information systems seek to provide specific and customized responses to individual user requests. Such information systems (IS) adapt their responses to user preferences, goals and capacities, in an effort to supply all the information required by users, and *only* the information required. This class of user-adaptive software systems is also called *information services* [4].

Thus, designing an IS for personalized information necessitates a user-centered view of the way the system manages and processes the data (e.g. the active part of the system). In addition to incorporating features that are common to all IS, the systems must also be able to adapt the results to a specific user. The relevance of the delivered results depends on several elements: the ability of the system to access and select the necessary data; the knowledge the system has about its users and its ability to learn so that this knowledge can evolve; and the pertinence of the retrieved data to the users' needs. These three elements are the primary features of information personalization.

The agent domain can provide the means for adapting results to system users. Software agents have already proved their ability to offer interesting services in uncertain, dynamic and open environments [12]. For example, agents can be service providers based on artificial intelligence techniques, such as machine learning. These agents can intervene at various stages in the modeling, design and implementation of an information system, or can simply serve as "concepts" or models for system analysis. In addition, they can be implemented jointly as a set of interacting agents, working together in what is called a multi-agent system.

In this article, we introduce MAPIS (Multi-Agent Personalized Information System), a multi-agent system for customizing information. Our objective is to provide models that can be reused for information system engineering, both at the macro level (e.g. the architecture, organization, and/or interaction between system entities) and at the micro level (e.g. knowledge representation, the agents' internal models). In the last section, these models are applied to travel planning services.

2 Information personalization in agent-based information systems

The main functions of an *information agent* have been summed up by Klusch [13]: information acquisition and management, information synthesis and presentation, and intelligent user assistance. Separately, each of these functions designates one area in the design and development of the *intelligent* aspects of information personalization, as well as one area of application for agent abilities. Moreover, in the context of multi-user IS and distributed data sources, the

interaction of the information agents can be managed in the frame of Multi-Agent Systems (MAS). The use of both IS approaches—agent and multi-agent—is presented below, with a specific focus on information personalization.

2.1 Agent abilities to personalize information

Agents possess several interesting characteristics in terms of information system design, including:

- *proactivity*, which allows the triggering of actions that have not been explicitly requested, meaning, for example, that a warning can be activated if an agent receives information that it deems useful for some users;
- *uncertainty management,* which is a key feature in Artificial Intelligence that allows agents to infer from their current incomplete knowledge and past experiences, making assumptions to compensate for lack of knowledge and/or learning from previous user transactions;
- *autonomy*, which allows agents to deal with distributed data and knowledge or processing resources; and
- *social abilities*, which allow agents in multi-agent systems to perform tasks requiring interaction between distributed entities, including knowledge sharing and task coordination.

These four characteristics can provide additional functions for an IS, principally in the three areas mentioned in section 2: information retrieval, information filtering and user assistance.

Information research and retrieval are dedicated tasks that can be performed by software agents. Such software agents make accessing information sources easier for users, for example via request refinement [16]. In one of the most well-known personal assistance systems, Letizia [18], an agent anticipates Web searches, recommending potentially interesting pages by deducing user interests based on the content of currently accessed pages. Agents can also regularly and proactively check data sources in order to warn users of eventual modifications. Such surveillance techniques makes it possible to “push” information, as in the case of a technological watch, for example [1].

Information filtering—the selection of relevant information in order to limit volume—is another task that can be accomplished by agents, with the selection based on user profiles defined in the filtering rules. Two key complementary methods are used: the cognitive method and the collaborative method. The first one simply analyzes document content [14]. The second one involves gathering user profiles showing similar interests. (See [13] for a survey of the principal methods.) The collaborative method is used, for example, to recommend new links, documents or products to users, based on statistical data about the choices of previous users. (See [28] and [7] for more information.)

User assistance draws on several different agent abilities in order to aid users in their individual tasks via an interface. This can mean adapting the hardware and software, a procedure that has become more and more important as the use of wireless information devices increases [3] [19]. It can also be a question of adapting the presentation to the user's preferences, with assistant agents observing and analyzing user actions on one or more software elements in order to automate some tasks [15]. Two design trends can be distinguished, both of which aim to automate certain tasks[24]: 1) a trend towards agent specialization in order to provide a specific service, such as sorting electronic mail or managing meetings; and 2) a trend towards the association of one agent and one user whose activities are well-known, in an effort to automate or delegate some tasks.

In order to assemble all of the above functions, one approach would involve a complex design that assigns the functions to a series of very “clever” agents possessing many skills and capacities. Another approach would be to distribute the functions to distinct agents in order to increase the flexibility and the adaptivity of the systems. In the latter approach, several (more or less) specialized agents are brought together to create a multi-agent information system.

2.2 Multi-agent information systems

The agents’ ability to both analyze user needs and to retrieve, understand and act on distributed data and knowledge about users is fundamental to agent-oriented information system design . However, further agentification of the system is required in order for information to be exchanged cooperatively. This agentification, which uses agent-oriented concepts to model some parts of the system, can be done to various degrees, ranging from the simple encapsulation of existing software elements to a complete agent-oriented analysis [31]. When designing an IS as a multi-agent system, each acting element is either an agent or a group of agents, whose interaction allows the information system to function. In multi-agent IS, the agents have organizational knowledge about each other’s competencies in order to answer questions, like *"Which agent can perform this task?"*, and to manage the data flows between agents, "W*hat must be done now?"*. This multi-agent approach can be tricky in that it is not only necessary to manage knowledge distribution and task distribution, but also to provide an efficient global process that will keep the agents organized.

Both knowledge and task distribution are more or less common to all multi-agent IS, depending on the three functional areas mentioned previously (sources, process, users). The agents’ abilities reproduce this functional decomposition, as shown in the three-layer architecture proposed by Shakshuki et al. [27]. Interface or assistant agents are used to manage agent-user interaction; information agents are used to collect data. Another type of agent processes the data and

matches it to user requests, and in fact, systems can be differentiated by the way in which this agent performs its task. The InfoSleuth [20] system, for instance, is dedicated to information gathering and complex query processing. It focuses on information semantics, with an in-depth use of ontologies. Another cooperative multi-agent information system, the PROFILE system [29] is also designed to personalize information, but its central focus is information discovery, based on domain knowledge.

As in the functional decomposition above, dedicated agents can also be used to coordinate the actions in the system. For example, Control agents create and manage the teams of agents in the MAPWEB system [6]. On the other hand, the agents in the ABROSE system [6] are completely self-organizing. Between these two approaches, there are systems composed of communicating cooperative agents that exchange knowledge and data. (See for example [27] and [29]). In every of these multi-agent organizations, the infrastructure of the system must allow concurrent agent activity and interaction, as is the case in RETSINA [30]. RETSINA enables agents with different competencies to interact and solve problems collectively. Its architecture emphasizes the structures required for agents to cooperate, plan actions, and decompose/recompose and delegate tasks, making it possible to create multi-agent systems able to perform all of these functions.

Though specific to information personalization, our approach is consistent with approaches that consider information systems as sets of organized, interacting agents. Our agents possess knowledge about users or information sources, thus allowing them to gather information and process complex queries. In this way, the MAS design for personalizing information is at the core of our system. Our goal is comparable to the PROFILE system, but the process is different. Our system, which focuses more on the management and use of user profiles than does the PROFILE system, is also comparable to system described by Shakshuki et al., with its three design levels. However, the objective of our middle level is closer to problem solving than to the matchmaking of the Shakshuki system. The knowledge distribution among the agents is also a bit different from other systems. For example, information sources in our system are represented as knowledge possessed by one or more agents, whereas the MAPWEB system is based on the direct association of one information source and one agent.

In the end, the goal of our approach is not so much to improve learning or filtering methods, but rather to provide reusable models for the system architecture, the agent types that compose it, and the coordination method. In this way, the models will be sufficiently application-independent to allow them to be used in different information domains.

3 Architecture and models for a multi agent-based information system

Modeling a multi-agent information system requires that several conceptual levels be described (see table 1). In our system, the highest level refers to system architecture and components; the intermediary level describes the interactions among the agents and agent groups that form the system; and the lowest level provides details about the agents' internal models.

Table 1: Conceptual models for the multi-agent system

Conceptual levels	Models	Some possible representations
Organization	Groups, roles, overall task processing	Class diagrams, sequence diagrams
Interaction	Communication and coordination processing	Sequence, interaction or activity diagrams
Behaviour	Competence handling, knowledge processing	State or activity diagrams

UML, specifically Agent-UML[1] (also called AUML [2]) is used to describe the models. This UML support has the advantage of providing some degree of standardization as well as allowing movement from a graphic representation to the more formal model semantics.

3.1 Multi-agent architecture

MAPIS, the Multi-Agent Personalized Information System that we propose, provides personalized access to an information set, by interacting with both users and information sources. (Figure 1: MAPIS interacts with the users via Assistant agents and with the databases via Search agents).

According to the ontology described in AUML [2], MAPIS is a *group employing* four types of *roles,* each played by different system agents (Figure 2):

- Assistant agents mediate between the users and the system,

[1] UML notation with extensions to represent the agents and their interactions. A number of research projects are working to extend UML in order to integrate agent specificities. See, for example, the introduction to notations adapted for mobile agents in M-UML [21].

- Search agents seek out data at its sources,
- Profile agents manage the user model, and
- Solver agents coordinate the information retrieval and personalization processes and integrate the data to generate an appropriate solution.

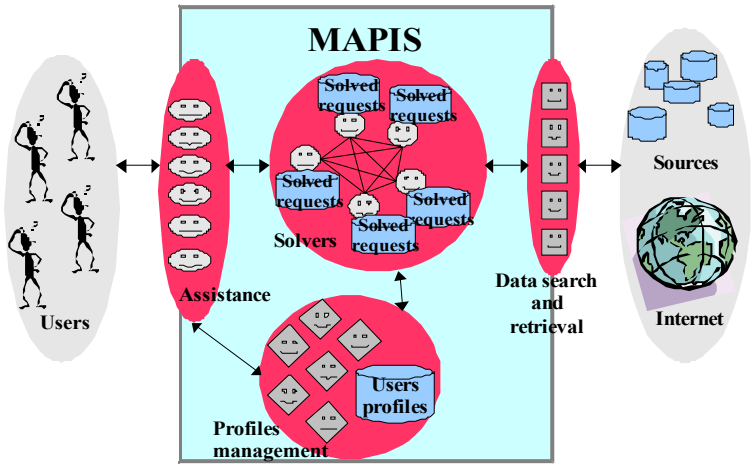

Figure 1: General description of MAPIS.

Assistant and Search agents come into play at the interface between the system and the external actors, users and data sources, while Solver and Profile agents are internal to the system.

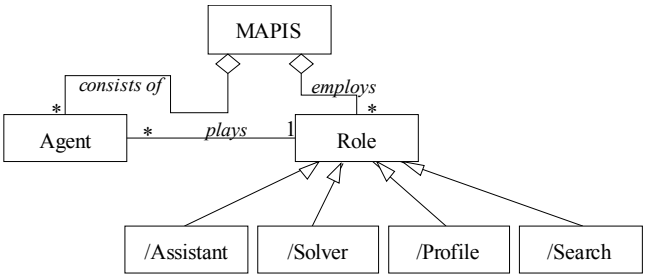

Figure 2: MAPIS agents and roles.

MAPIS is made up of a set of interacting agents. Each agent plays a role corresponding to certain specific abilities, and each role is played by several agents. (The Agent-Role association is shown in Figure 2.) Thus, the system is essentially composed of four teams of specialized agents, with each team playing one role. We chose to deal only with the case of specialized agents because a system of multi-competence agents would make the agent code more complicated and would multiply the number of messages exchanged in the search for a competent but idle agent.

3.2 Interaction

The interaction among the system agents permits the delegation of tasks and the transmission of user request and profile data, as well as retrieved data. The communication between the agents is based on a protocol that is defined according to the specific needs of the multi-agent system.

3.2.1. Communication protocol requirements

Communication between the agents is accomplished via message exchanges. It is necessary to have point-to-point contacts between two identified agents, as well as multicast contacts for sets of agents playing a given role. In addition, the agents must also be able to use both synchronous and asynchronous blocking exchanges, in order to facilitate continuous agent activity. For example, the search for an agent with specific competencies, such as the ability to interact with the user, requires the diffusion of a multicast asynchronous message among agents playing the Assistant role. Idle agents answer the sender via a synchronous message, in order to maintain availability as long as needed in order to conclude the exchange. The agent that initiated the communication confirms its agreement to the first responding agent in order to conclude the transaction and sends its disagreement to the others to release them.

An illustration of an exchange of this type is given in Figure 3 a), in which an agent *A1* sends a request message to *m* agents having a specific role. The *n* idle agents respond with an availability message and are retained. The *A1* agent

gives its consent to the first responding agent (called Participant in the figure) and releases the others, making them idle once again, and thus free to contract other tasks. If no agent playing the required role is idle, the initial call is repeated. The Participation search protocol depicted in Figure 3 a) is based on the FIPA Iterated Contract-Net [8]; Figure 3 b) presents another protocol, called a Similarities search. In this protocol, the initiating agent sends a call for other agents that have previously solved a problem similar to the current one. Both the Participation search and the Similarities search are frequently used in communication and synchronization schemes between the agents, and thus are presented as sub-diagrams in order to avoid repetition in the more general models.

The content of the messages depends on the type of message (request, answer, etc.) and the data exchanged concerning the retrieved information and the user profile. The communication protocol is based on four types of messages:

- *cfp*: call for participation in a task
- *cfs*: call for similarities between past and current tasks
- *answer*: answer a call (accept or refuse)
- *inform*: data transmission

Six data types are possible, depending on the type of message:

- *subscription*: subscription data
- *ident*: identification information to be transmitted to the user
- *request*: a user request
- *profile*: a user profile
- *info*: retrieved information
- *result*: the final information for the whole process

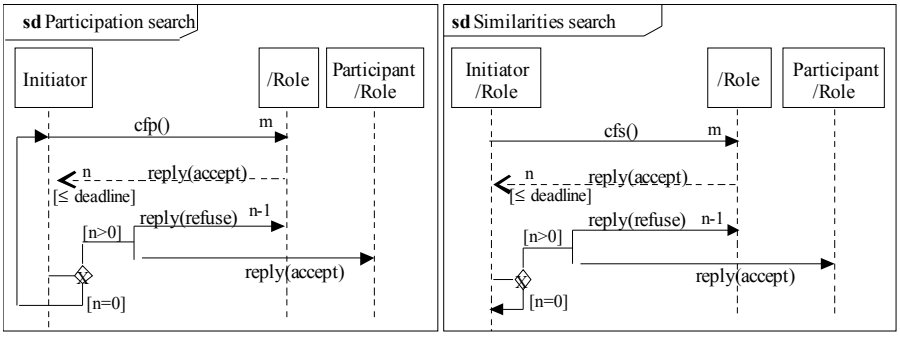

Figure 3: Elementary interaction sub-diagrams (sd) for MAPIS agents.

3.2.2 Personalization process

The MAPIS multi-agent system bases its personalization process on the management and processing of data about the information domain, the history of requests and responses, and the users.

Let D be the data set defining the domain of the information system.
Let D' be the data set contained in a request form. D' is not the subset of D , but D' is dependent on D.
For example:

- if D corresponds to the trip planning information, then D' ={departure; arrival; hour_arrival; hour_departure; date; reason_trip}
- if D corresponds to the musical base, then D'={title_album; title_song; date; interprete}

Let R be the set of possible requests. A request is a n-uplet, such that:

$$\forall r \in R, r = \langle d'_1, d'_2, \ldots, d'_n \rangle \text{ with } d'_i \in D'$$

Let A be the set of possible results. A result is the set of all the possible solutions for a given request, with a given solution being a k-uplet, such that:

$$\forall a \in A, a = \{A_j\} \text{ with } A_j = \langle d_1, d_2, \ldots, d_n \rangle \text{ and } d_k \in D$$

The users are modeled according to the characteristics required to complete the personalization.
Let U be the set of known users, Q the set of criteria and preferences, as defined by the designer according to the information domain.

The knowledge the system has about the users is divided into three data sets: the static data set given to the system by the user (Su); the data set deduced by the system (Pu); and the data set resulting from the history of user-system transactions (Hu).

$\forall u \in U$, the profile of the user u is $M_u = (S_u, P_u, H_u)$

Let Su be the static data set required to identify and alert the user u.
Let Pu be the weighted data set necessary to personalize the information, such that $P_u = \{(x_q, y_q), \forall q \in Q\}$ where

- x_q is the importance value between 1 and α
- y_q is the confidence value between 1 and α

By default, the x_q weights are set to the value α/2 (q is a criterion of average importance) and the y_q weights are set to the lowest value, 1 (indication the low confidence of the system with regard to criterion q).

Let H be the set of solved requests, such that $H_u = \{(r, a, \omega), r \in R, a \in A_r\}$, where ω is a function defining the rang of a in A_r (the set of the possible solutions for the request r).

The personalization process can be described using the above definitions. Figure 4, which presents the activity transfer between the roles, depicts a generic approach to the overall interaction process. (See [22] for a more detailed description of the personalization methods.)

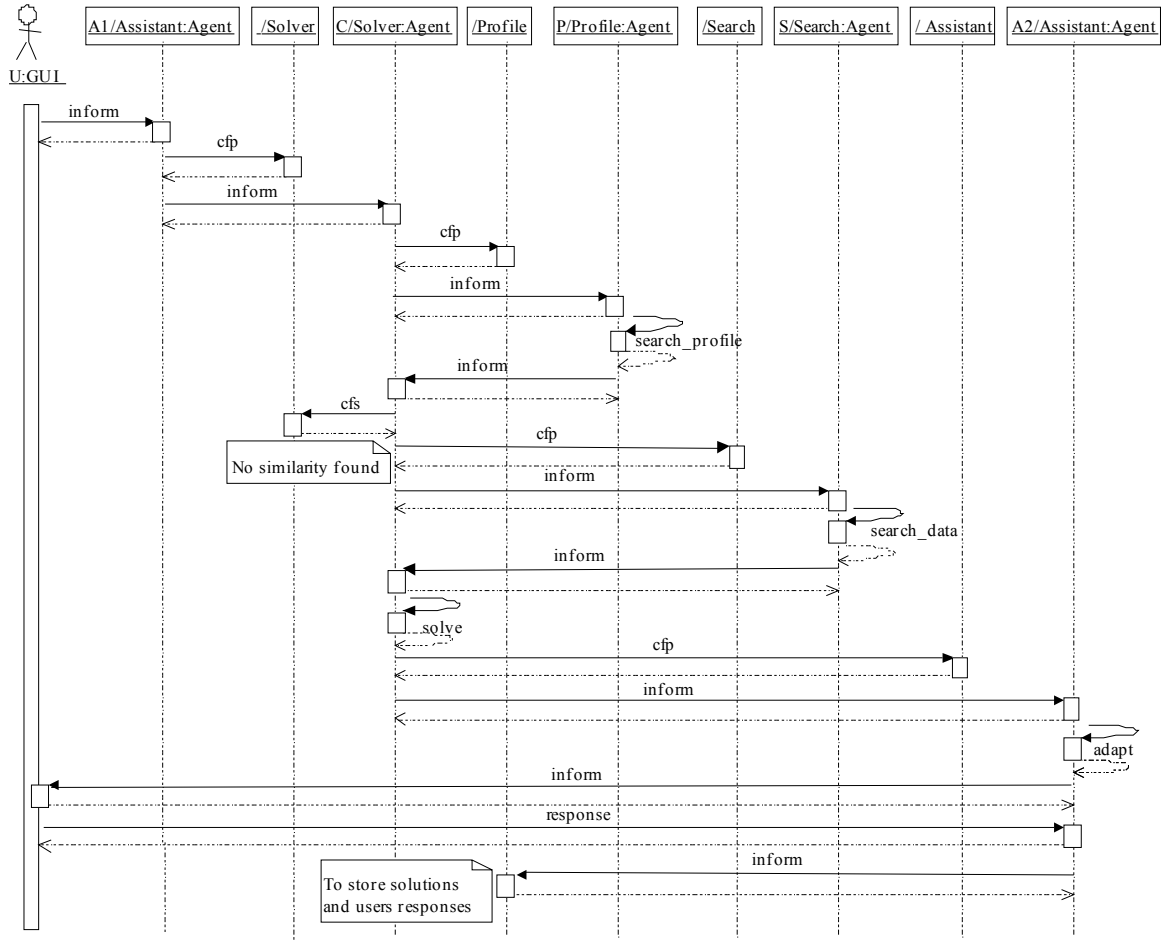

Figure 4: The personalization process.

The above diagram presents an example of the overall interaction process:

- User U searches for information (R) via the GUI (Graphical User Interface). This request (R) is transmitted to A1 (an assistant agent).
- A1 transmits R to C (a solver agent).
- C needs User U's profile to determine the best solution, and so asks P (a profile agent) to provide it.
- Before searching for the relevant data, C checks its database to see if any similar data exists (similar data corresponds to an identical request from a similar profile).
- Then, C sends a call for similarities to all the other solver agents.
- If there is no positive response, C determines the best solution by asking S (a search agent) for the relevant data for R and calculating a solution.

This solution will be presented to U via A2 (another assistant agent).
This diagram (Figure 4) is representative of instances in which the Solver agents have no similar requests in their respective knowledge bases, as well as those in which the user refuses the first solution but accepts the second one.

The personalization process itself is divided into sets of distinct tasks, each requiring the specific skills of the agents. Each given task type corresponds to a set of required skils and a specific agent role. Thus, the process involves delegating tasks from one agent to another, every time a task requires skills that the initially contacted agent does not possess. Assistant agents carry out distinct activities according to the type of request received, either a subscription or a request for information. They transfer activities towards both the Profile and the Solver agents. The Solver agents work on the simultaneous tasks of data profiling and data retrieval. They delegate the related tasks to the Search and Profile agents, creating sub-groups of two or more agents that cooperate at a given time.

Agent-to-agent task delegation includes both a search for an agent able to pursue the process and a transfer of the activity and the data to the agent that is found. This delegation process repeatedly exploits the elementary *Participation search* protocol described in section 3.2.1 (Figure 3). This protocol both ensures fundamental agent autonomy and facilitates agent resource management. The Solver agents, on the other hand, use the *Similarities search* protocol to exploit past experience, looking for previous similar requests inside the Solver team. These agents use another type of activity transfer. By calling for agents that have solved similar requests in the past, they can reuse previous solutions. Note that the term "similar" is used both for requests and for profiles, though the meaning is slightly different. Requests are said to be similar when they are identical and associated to similar profiles. Profiles are said to be similar when the difference between the weight of each criteria is lower than delta (a value defined by the system administrator).

$$(\mathrm{sim}(P_{u1}, P_{u2})) \Leftrightarrow (\forall q \in Q, \forall (x1_q, y1_q) \in P_{u1}, \forall (x2_q, y2_q) \in P_{u2}, (x1_q = x2_q) \wedge (|y1_q - y2_q| < \delta))$$

In summary, the personalization process is based on a set of interactions between agents. Agents contact one another, perform the required tasks, and become idle again. Each time an agent handles a new request, it initiates a sequence of sub-group creations, formed around the required personalization tasks. The organization is dynamic; at any given moment, the system can be composed of different and variable sub-groups, or *teams,* whose members share a common goal.

3.3 Agent models

As mentioned previously, each role requires specific skills and is linked to a distinct activity cycle. Each agent has some basic skills common to all the agents (e.g. the communication protocol, the *cfp* protocol) and some specific skills related to its role, as well as certain knowledge sets that are also linked to its role: A=(BasicSkills, SpecificSkills, KnowledgeSet). It follows, then, that each agent model in our approach is defined according to its role.

Let As, Se, Pr and So be the Assistant, Search, Profile and Solver roles. Let A be the agent set, such that As, Se, Pr, and So are subsets of A.

The Assistant agents are responsible for the interaction between the system and the users. They receive the data from the users, analyze it and transmit it to the system's internal agents. They also receive result data from the other agents in order to present it to the users. Two types of input are thus defined: requests and answers. A simplified representation of the Assistant agent's activity is presented in Figure 5.

Assistant agents must have the ability to analyze user requests (*verif* function) and to present answers in a format that suits the user's preferences (*adapt* function). These functions can be carried out at various levels of complexity. For example, the request analysis can use a simple parser or can employ more complex ontologies to enrich the semantics [17]; the results presentation can be adapted to reflect simple changes in the graphic format or can require more complex notions, such as interface plasticity [5].

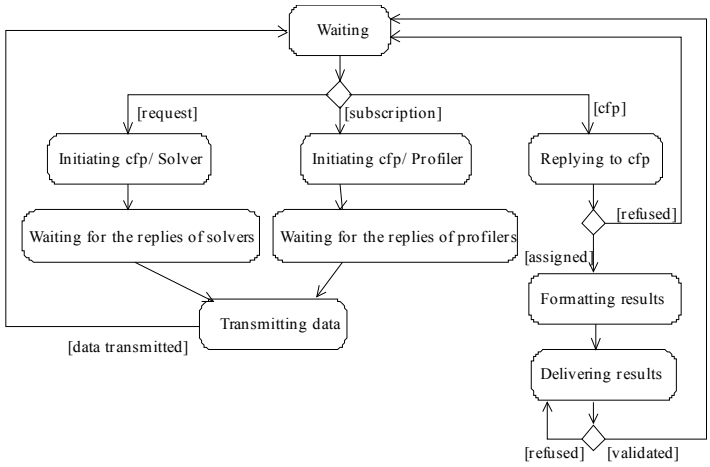

Figure 5: The Assistance agent activity cycle.

The Assistant model of MAPIS is defined below:
As=(BasicSkills, SpecificSkills, KnowledgeSet) , where

$$KnowledgeSet = \{\{d', d' \in D'\} \cup skills(Solver) \cup skills(Profile)\}$$

$$AssistantSkills = \{verif : R \rightarrow R'\}$$

While the Assistant agents manage user-system interaction, the agents playing the Search role intervene at the interface between the system and the external data sources. They receive information demands and transmit the retrieved information (Figure 6). Their required skills are linked to data access, specifically in terms of mobility. They search for the appropriate data (*search_data* function).

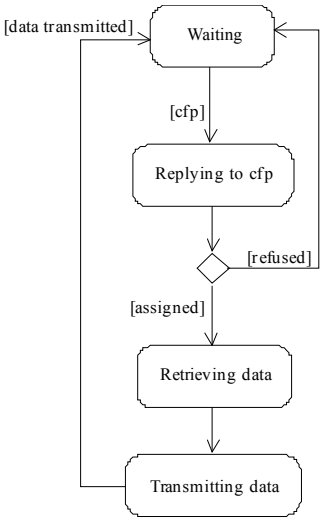

Figure 6:The Search agent activity cycle.

The Search model has been defined as follows:
Se=(BasicSkills, SearchSkills, KnowledgeSet) , where

$$KnowledgeSet = \{d, d \in D\}$$

$$SearchSkills = \{searchData : R \rightarrow P(D)\}$$

The agents playing the Profile role create, manage and update user profiles. They create new profiles for new subscribers to the system, answer information requests from other agents with regard to users, and update existing profiles to reflect the new cases introduced in the connection history (Figure 7). The main skill of Profile agents is the ability to learn, regardless of the technique used to update the profiles (*update* function).

Our approach employs stereotypes to instantiate user profiles the first time a user connects to the system. Then the user's characteristics are adjusted using two weights: an importance weight and a confidence weight. This weighting makes it possible both to assign relative importance to the characteristics and to take the confidence of the system in this importance rating into consideration. Given that some of the relative importance weights are the result of

deductions based on past uses of the system, the confidence weighting provides some knowledge about the deduced importance weights. The more frequent the user requests and the more consistent the answers, the more confident the system is in the results that are given. Both weighting values are adjusted by the *update* function.

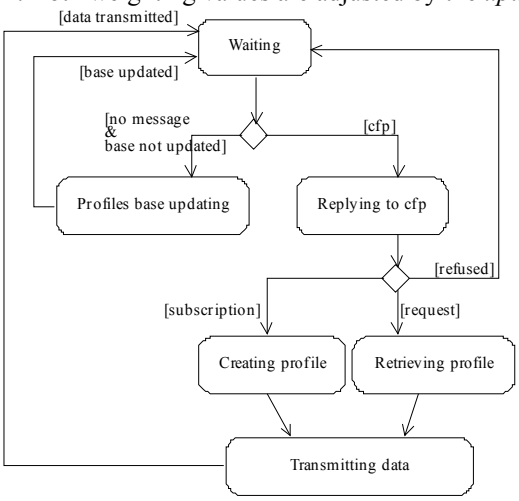

Figure 7: The Profile agent activity cycle.

The Profile model is defined as follows:
Pr=(BasicSkills, ProfileSkills, KnowledgeSet) , where

$$KnowledgeSet = \{ M_u, \forall u \in U \}$$

$$ProfileSkills = \{ searchProfile : U \to M_u, update : R_u \times H_u \to R_u \}$$

Finally, the agents playing the Solver role initiate information research and information personalization with the help of the other agents. They have the central role in the data retrieval process of as it is these agents that generate an appropriate solution (*solve* function) (Figure 8). Essentially, these agents must have reasoning abilities so that they can integrate the data and coordinate the other agents' activities.

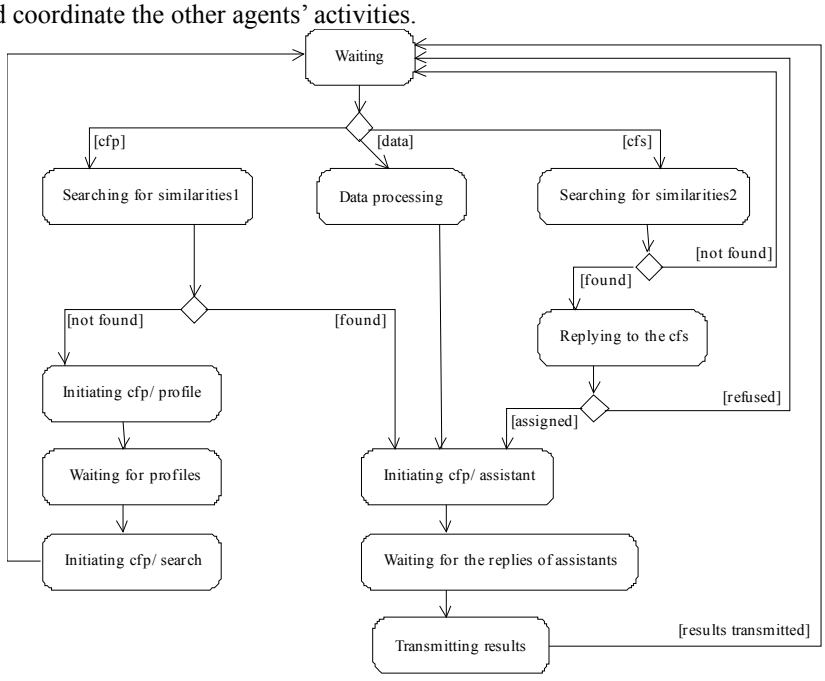

Figure 8: The Solver agent activity cycle.

The Solver model is defined below:
So=(BasicSkills, SolverSkills, KnowledgeSet) , where

$$KnowledgeSet = \{ (h_u, P_u), h_u \in H_u \}$$

$$SolverSkills = \{ solve : R \times R_u \times P(D) \to A, cfs : R \times R_u \to S_o \}$$

The *cfs* function yields the name of the solver agent that is able to solve the request, thanks to its knowledge base.

Each of these four models contribute to the personalization process (Figure 9):

1. Selection of the databases, done by the search agents in order to decrease the information flow.
2. Management and update of the user profiles, done by the profile agents in order to retain relevant knowledge about each user.
3. Determination of the match between the set of data and the user profile, done by the solver agents in order to supply a relevant response according to user preferences and/or needs.
4. Adaptation of the response, done by the assistance agent in order to conform to the user's desired graphic interface.

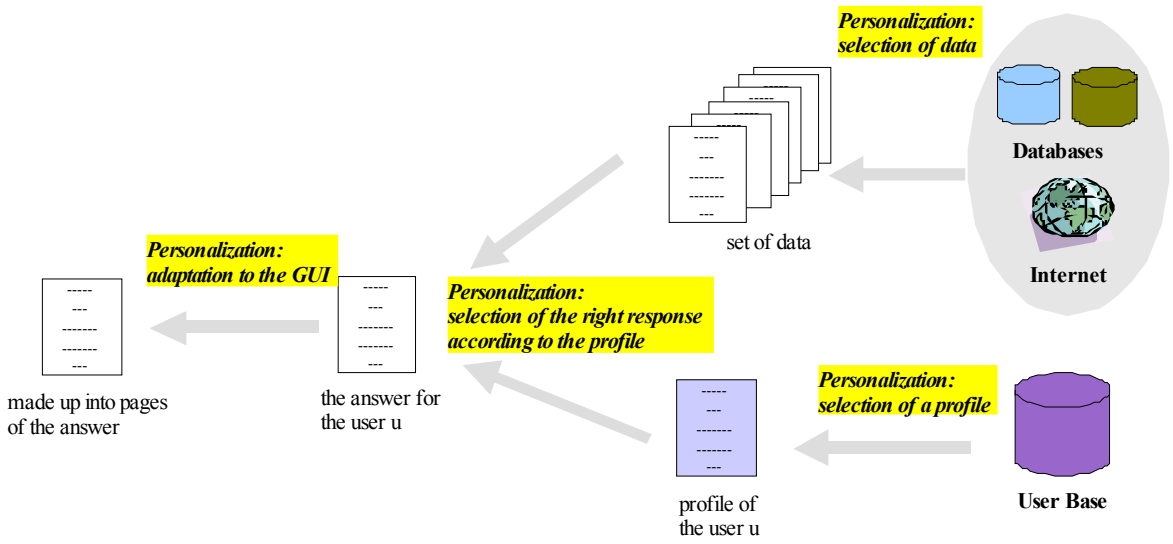

Figure 9: The steps of the personalization process.

It is possible to skip any of these steps; however, since each step (or each agent model) supplies a part of the personalization, removing any of them will result in a less personalized final response. The set of the personalization actions aims to supply an optimal personalized answer.

4 Application of MAPIS to AgenPerso, a personalized transportation information system

The MAPIS multi-agent organization is currently being tested in a pre-trip travel information system, called AgenPerso. Created as a prototype personalized information system integrating multiple types of transport, AgenPerso implements MAPIS in a defined applicative context, thus allowing our decomposition technique for information personalization to be verified and validated.

Following a feasibility study examining the benefits of using agents for information systems involving multiple types of transport [23], we implemented MAPIS in the pre-trip travel information system, AgenPerso [21], which is designed to provide personalized help for travel planning. Unlike other travel planning tools, including those based on multi-agent systems [6], this MAPIS-based system permits the personalization of information. If necessary, the travel plan can include a set of transportation modes (e.g. bus, subway, taxi, walking) designed to satisfy the needs, capabilities and preferences of each user as much as possible. Though the application is still at the prototype stage, it is possible to generalize about the type of personalized answers that the software agents can generate.

AgenPerso required the following adaptations for MAPIS implementation.

- The Assistant class was instantiated after filling the agents' knowledge bases with information related to the reasons for travel. The system does, in fact, ask the travelers to indicate the event for which they must travel, for instance "attend the play, Hamlet" (one "event" item is chosen instead of an Arrival date or place, using a request form).
- The Solver class was implemented after an overload of the solution selection method. Implementing the Solver class requires knowledge of the criteria used to select a personalized answer from all the possible solutions. The process of applying the data depends on the expected functions of the system: simple filter, information gathering or a more specific processing, such as planning or problem solving. In the context of transportation information, the best solution satisfies a classic criteria set including transportation modes, durations and costs.
- The Search class agents must know which data sources to use, the way to access them, their data format, etc. The data can be stored in the IS or can be accessed through a local or an external net. In the latter case, the agents can be

provided with the ability to find the appropriate sources themselves, as in Shakshuki et al. [27]. In our application, the Search agents were implemented as soon as we could provide the knowledge about the data sources, e.g. the access to nets and timetable databases.

- Instantiating the Profile class required preliminary work. In fact, several questions had to be asked: Which stereotypes can be distinguished? Which are the most appropriate learning techniques to use (Q-Learning, observation-based [10], etc.)? Which parts of the requests and the answers must be stored? In addition, since the knowledge about the user must be defined, namely the fixed and the dynamic data which are required by the system, other questions arose, for example: Is it necessary to know the user's address or the user's age? AgenPerso required that two modules be defined: a module to record and sort the *requests/answers* couples that were accepted by the users, and a knowledge-based reasoning module to modify the weighting of the selection criteria on the basis of answer statistics.

Because the personalization process is based on the knowledge that the system has about the user, each new user must start by completing a subscription form, which ask for three types of information:

- Information that allows identification or contact if necessary: e.g. address, phone number, etc.,
- Information that allows the attribution of a default profile: e.g. a stereotype selection, such as a professional category,
- More detailed information about user preferences about transportation: e.g. preferred walking distances.

On the base of these data, a Profile agent creates an initial profile for the user (see Figure 10.a). The stereotyped values are then altered to fit this specific user (Figure 10.b).

(a)

```
<User>
  <Access>
    <Login>mrdub</Login>
    <Passwd>toto</Passwd>
  </Access>
  <GeneralBackground>
    <Person>
      <Name>Dubois</Name>
      <FirstName>Dubois</FirstName>
    </Person>
    <Mail>mrdub@wanafree.fr</Mail>
    <Birthday>4/april/1940</Birthday>
    <Address>Lilas street</Address>
    <Profession>Student</Profession>
  </GeneralBackground>
  <History/>
  <TransportPreferences context="general">
    <Criteria>
      <Economic Importance="6" Validity="1"/>
      <Quick Importance="5" Validity="1"/>
      <Corresp Importance="4" Validity="1"/>
      <Distance Importance="5" Validity="1"/>
    </Criteria>
    <ModeTransport>
      <Bus Importance="5" Validite="1"/>
      <Metro Importance="0" Validite="10"/>
      <Train Importance="5" Validite="1"/>
      <Tramway Importance="5" Validite="1"/>
      <Walking Importance="1" Validite="9"/>
      <Taxi Importance="5" Validite="1"/>
    </ModeTransport>
  </TransportPreferences>
</User>
```

A stereotypical profile

(b)

```
<User>
  <Access>
    <Login>mrdub</Login>
    <Passwd>toto</Passwd>
  </Access>
  <GeneralBackground>
    <Person>
      <Name>Dubois</Name>
      <FirstName>Dubois</FirstName>
    </Person>
    <Mail>mrdub@wanafree.fr</Mail>
    <Birthday>4/april/1940</Birthday>
    <Address>Lilas street</Address>
    <Profession>Student</Profession>
  </GeneralBackground>
  +<History>
  <TransportPreferences context="general">
    <Criteria>
      <Economic Importance="6" Validity="1"/>
      <Quick Importance="4" Validity="3"/>
      <Corresp Importance="5" Validity="3"/>
      <Distance Importance="5" Validity="1"/>
    </Criteria>
    <ModeTransport>
      <Bus Importance="7" Validite="1"/>
      <Metro Importance="0" Validite="10"/>
      <Train Importance="5" Validite="1"/>
      <Tramway Importance="5" Validite="1"/>
      <Walking Importance="1" Validite="9"/>
      <Taxi Importance="5" Validite="1"/>
    </ModeTransport>
  </TransportPreferences>
</User>
```

The preferences after the learning

Figure 10: Example of the adjustment of profile characteristic weights.

Two kinds of requests can be introduced: classic requests composed of departure and arrival dates and places, or novel requests that indicate the reason for travel by selecting an event in the list of the events known to the system. The system then plans the trip, taking into account all the possible known modes and the user's known particularities. For examples, Figures 11 and 12 present the results given for the same request entered by two different persons: a user whose primary criterion is financial; and another one whose first criteria include walking distance and connections. The system proposes the solution it judges to be the most pertinent from the set of possible solutions. The user can accept this solution or not. In the case of refusal, the system proposes the next solution, if one exists, in order of supposed pertinence.

Figure11: Examples of the request and result interfaces for a student user

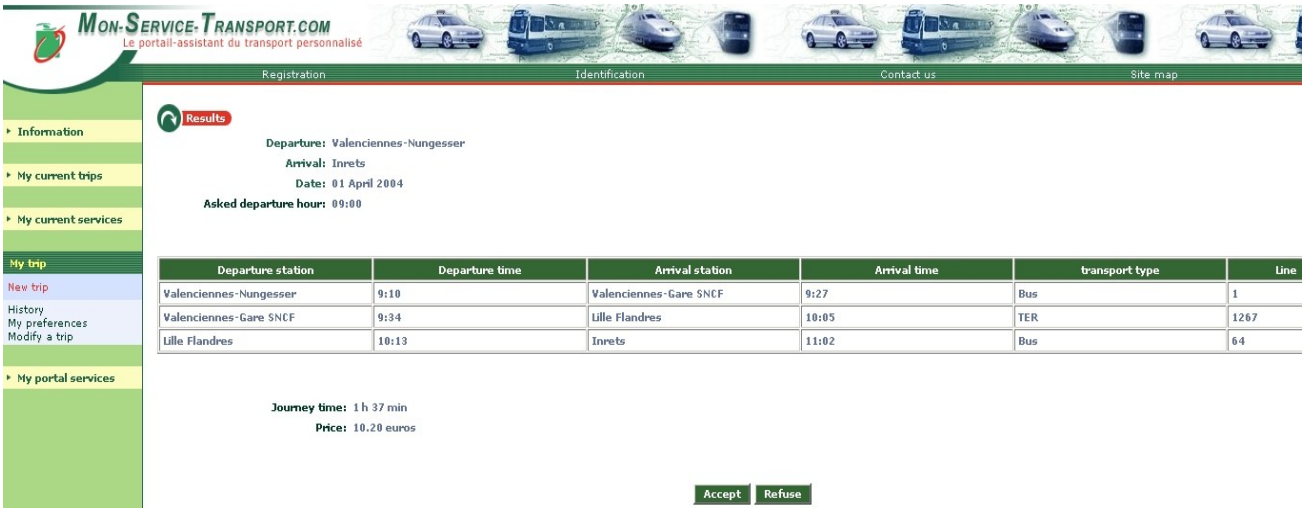

Figure12: Examples of the request and result interfaces for an elderly user.

The application provides an example of how MAPIS can supply the models that are required to implement multi agent-based information personalization. The agent classes have been instantiated and detailed in order to fulfill the application domain requirements. The knowledge included in the user model and the personalization criteria must be specified in the information system application. On the other hand, the distribution of the four agent roles, the agent activity cycles, the *call for participation* and *call for similarities* protocols, make the overall personalization process independent from the application.

5 Conclusion and future works

Multiagent-oriented IS are developed as multi-agent systems from the first step of their design up until their implementation. Their design requires the use of agent models as a basis for the organization, interaction and behavior levels. We have proposed a set of these models as part of the design for a multi-agent system called MAPIS for use in an information system that delivers personalized responses. The overall information process is divided into tasks requiring specific skills: interaction with the user, interaction with data sources, user profile management and information processing. The skills are grouped according to four roles, which are the four classes of agent behavior present in the system.

The next step in this research will involve studying a design method and proposing a tool that supports the design of personalized information systems based on MAPIS. A Computer-Aided Software Engineering (CASE) tool would offer design, integration and development services from the agents' pre-established models, thus assisting designers by

allowing them to focus on other problems, such as the definition of ontologies related to the precise application domain of the system. The use of patterns that include models, behaviors and code generation (like those described by Weiss [32], for example) would be one way to achieve the reusability objective.

Our trip planning prototype needs to be validated with more data and a greater number of users. We would like to make it possible to connect the system to personal agendas (for example PDA) in order to allow additional information to be taken into account, as well as to provide users notification in the event of changes which would affect their trip. The system would allow the integration of travel-related data, such as accommodations and catering obviously, but also such information as opening times for shops or public administrations. Another improvement would be to make it possible to obtain information in different formats in order to adapt the presentation to the hardware/software available to the user: personal computers, public information points or mobile phones, for instance.

Acknowledgements
This research was partially financed with grants from the PREDIT program.

References

[1] E. Adam, R. Mandiau, C. Kolski, Application of a holonic multi-agent system for cooperative work to administrative processes. Journal of Applied Systems Studies, 2 (2001), pp. 100-115.

[2] B. Bauer, J.P. Muller, J. Odell, Agent UML: A formalism for specifying Multiagent Interaction, in: P. Ciancarini, M. Wooldridge (Eds.), Agent-Oriented Software Engineering, Springer-Verlag, Berlin, 2001, pp. 91-103.

[3] D. Billsus, M. J. Pazzani, J. Chen, A learning agent for wireless news access. Proceedings of 2000 International Conference on Intelligent User Interfaces, New Orleans, USA, (2000), pp. 94-97.

[4] P. Brusilovsky, Adaptive hypermedia. User Modeling and User-Adapted Interaction, 11 (2001), pp. 87-110.

[5] G. Calvary, J. Coutaz, D. Thevenin, Q. Limbourg, N. Souchon, L. Bouillon, M. Florins, J. Vanderdonckt, Plasticity of User Interfaces: A Revised Reference Framework, in: Proceedings of the First International Workshop on Task Models and Diagrams for User Interface Design. Inforec Publishing House Bucharest, 2002, pp. 127-134.

[6] D. Camacho, J.M. Molina, D. Borrajo, A multiagent approach for electronic travel planning, Second International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2000) at AAAI-2000, Austin (Texas, USA).

[7] G. Carenini, J. Smith, D. Poole, Towards more conversational and truly collaborative recommender systems, Proceedings of the 2003 International Conference on Intelligent User Interfaces, ACM Press, 2003, pp. 12-18.

[8] FIPA: Foundation for Intelligent Physical Agents (2002). FIPA Iterated Contract Net Interaction Protocol Specification. Document SC00030H, December. Available at: http://www.fipa.org/specs/fipa00030/SC00030H.pdf

[9] M.P. Gleizes, P. Glize, J. Link-Pezet, An Adaptive Multi-Agent Tool For Electronic Commerce, The workshop on Knowledge Media Networking, 9th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2000), 4-16 June, Gaithersburg, MD, USA. IEEE Computer Society, 2000, pp. 59-66.

[10] J. Goecks, J. Shavlik, Learning Users' Interests by Unobtrusively Observing Their Normal Behavior, Proceedings of the 2000 International Conference on Intelligent User Interfaces, ACM press, 2000, pp. 129-132.

[11] O. Gutknecht, J. Ferber, F. Michel, The Madkit Agent Platform Architecture, Research Report, LIRMM, Montpellier, France, 2000.

[12] N. R. Jennings, On Agent-Based Software Engineering, Artificial Intelligence, 117 (2000), 277-296.

[13] M. Klusch, Information agent technology for the Internet: A survey. Data & Knowledge Engineering, 36 (2001) 337-372.

[14] B. Krulwich, C. Burkey, The InfoFinder agent: Learning User Interests through Heuristic Phrase Extraction, IEEE Expert: Intelligent Systems and Their Applications, 12 (1997) 22-27.

[15] Y. Lashkari, M. Metral, P. Maes, Collaborative Interface Agents, Proceedings of the Twelfth National Conference on Artificial Intelligence, vol. 1, Seattle, USA, AAAI Press, 1994, pp. 444-449.

[16] W.P. Lee, T.C. Tsai, An interactive agent-based system for concept-based web search, Expert Systems with Applications, 24 (2003) 365-373.

[17] M. Li, M. Qi, MAPBOT: a Web based map information retrieval system, Information and Software Technology, 45 (2003) 691-698.

[18] H. Lieberman, Letizia: An Agent That Assists Web Browsing. International Joint Conference on Artificial Intelligence, Montreal, Canada, Aug. (1995)

[19] Moraitis, P., Petraki, E. and Spanoudakis, N.. "Providing Advanced, Personalised Infomobility Services Using Agent Technology". 23rd SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence (AI2003), Peterhouse College, Cambridge, UK, 15th-17th December 2003

[20] M. Nodine, J. Fowler, B. Perry, An Overview of Active Information Gathering in InfoSleuth, Proceedings of the 2nd International Symposium on Cooperative Database Systems for Advanced Applications, Springer-Verlag, 1999.

[21] C. Petit-Rozé, A. Anli, E. Grislin-Le Strugeon, M. Abed, C. Kolski, AGENPERSO - Interfaces Homme-Machine à base d'AGENts logiciels PERSOnnels d'information aux usagers des TC, Final report of PREDIT project, Valenciennes, France, 2003.

[22] C. Petit-Rozé, E. Grislin-Le Strugeon, Intelligent Agents to Structure and to Process Personalized Information Systems, Proceedings International Conference on Intelligent Agents, Web Technologies and Internet Commerce, DiskTech Pty Ltd, Canberra, Australia,2003, pp. 165-174.

[23] C. Petit-Rozé, E. Grislin-Le Strugeon, Interaction with Agent Systems for Intermodality in Transport Systems, in: M. Smith, G. Salvendy, D. Harris, R. Koubek, Usability Evaluation and Interface Design: Cognitive Engineering, Intelligent Agents and Virtual Reality , LEA, London, pp.494-498, 2001.

[24] M.P. Ramos, Structuration et évolution conceptuelles d'un agent assistant personnel dans les domaines techniques, PhD Thesis, France, 2000.

[25] S. Ravden, G. Johnson, Evaluating Usability of Human-Computer Interface: A Practical Method, Series in Information Technology, Ellis Horwood, Chichester, 1989.

[26] K. Saleh, C. El-Morr, M-UML: an extension to UML for the modeling of mobile agent-based software systems, Information and Software Technology, 46(2004) 219-227.

[27] E. Shakshuki, H. Ghenniwa, M. Kamel, An architecture for cooperative information systems, Knowledge-Based Systems, 16 (2003) 17-27.

[28] U. Shardanand, P. Maes, Social Information Filtering: Algorithms for Automating 'Word of Mouth', Proceedings of the CHI-95 Conference, ACM Press, 1995, pp. 210-217.

[29] J. Simons, A. Arampatzis, B.C.M. Wondergem, L.R.B. Schomaker, P. van Bommel, Th.P. van der Weide, C.H.A. Koster, PROFILE – A Multi-Disciplinary Approach to Information Discovery, in: G. Wagner, Y. Lespérance, E. Yu (Eds.), Agent-Oriented Information Systems 2000, Proceedings of the 2nd International Workshop at CAiSE*00, Stockholm, 2000.

[30] K. Sycara, J.A. Giampapa, B.A. Langley, M. Paolucci, The RETSINA MAS, a Case Study, in: A. Garcia, C. Lucena, F. Zambonelli, A. Omici, J. Castro (Eds.), Software Engineering for Large-Scale Multi-Agent Systems: Research Issues and Practical Applications, Springer-Verlag, Berlin Heidelberg, LNCS 2603, July, 2003, pp. 232-250.

[31] G. Wagner, Information Systems Have to Deal with Objects and with Agents, International Bi-Conference Workshop on Agent Oriented Information Systems, Heidelberg,1999.

[32] M. Weiss, Patterns for Motivating an Agent-Based Approach., in: M.A. Jeusfeld, O. Pastor (Eds.), Conceptual Modeling for Novel Application Domains, Springer-Verlag, LNCS 2814, 2003, pp. 229-240.