Facilitating User–System Interaction: the GAIA Interaction Agent

Authors: Koutsabasis P., Darzentas J. S., Spyrou T., Darzentas J.,
Research Laboratory of Samos, Department of Mathematics,
University of the Aegean, GR 83200 Karlovassi, Samos, Greece, email : kgp@aegean.gr

Abstract

This paper presents the identification, design and implementation of a user interface to a brokerage system and the conceptual architecture and functional behaviour of an intelligent interaction agent that supports and enhances the interaction between the user and the system. The term interaction agent is used in order to describe a particular class of interface agents that function as intelligent personal assistants to users of a computer - based system. The interaction agent provides assistance to the user in two contexts. On the user interface level, it assists users to comprehend and manipulate the user interface. On the domain of application level, it provides users with information and advice according to their preferences. This work is being carried out in the context of an ACTS (AC 221) project GAIA (Generic Architecture for Information Availability).

1. Introduction

The intelligent agent paradigm has gained much popularity in the last few years, although the concept was first discussed several years ago [28]. From the application point of view, there are agents that are invisible to the user. These could be broadly described as autonomous processes that perform well-defined tasks, such as network monitoring (SNMP MIB agents), alerting services [1], independent searching (search engines' spiders), etc. Such autonomous processes qualify as agents because they possess many of the characteristics that define agents, such as autonomy, mobility, social ability and others [10,17,11,32].

Other agents, on the other hand, are highly visible to the user. They take on the metaphor of agent in the real-world meaning of the term agent, as in travel agent, or estate agent, or personal assistant. These agents are sited at the interface of a computer system and «intervene between the human user and the computer system» [23]. It is claimed that user interface agents act as a bridge between a user's goals and the computer [38]. Such agents can make the interface more intuitive and can encourage types of interactions that might be difficult to evoke with a conventional interface. A variety of user interfaces that employ some sort of agency attributes have been termed 'interface agents', from Web search engines, to lifelike, animated characters. Considerable work has been done in the last few years with regard to interface agents. They have been mostly employed to assist users with their personal everyday tasks such as managing electronic mail [26], meeting scheduling, [26,15,5], personalised information filtering [26,27], recommendation systems [12,33,6,9], and others [17]. Such systems, while they emphasise the capability of the agent to utilise user preferences, mostly restrict their activity to tasks at the interface. That is to say, they undertake to personalise some part of the tasks the interface supports. In addition work on intelligent interfaces could also be mentioned in the context of user support at the interface [36].

The term `interaction agent` however [21], is used to describe an intelligent interface agent that supports the whole of the interaction between the user and the underlying system. An interaction agent interacts with both ends of the communication channel: user and underlying subsystem, and is capable of reasoning about inputs it gets from both sides of interaction. The interaction agent's twofold goal is to assist users in both manipulating the computer–based system, and providing users with information and advice with regard to application domain actions and options according to their preferences. In order to reason about its actions an interaction agent needs firstly, to employ a representation of user goals and knowledge of the domain. This enables the agent to observe user actions and to reason about their efficiency in order to guide the interaction when appropriate, thus helping users to perform tasks via the application user interface. Secondly, the interaction agent needs to possess its own knowledge of the application domain, enabling it to suggest, for instance, search terms

0-7695-0001-3/99 $10.00 (c) 1999 IEEE

to users who may have difficulty in expressing their requirements to the search and locate part of the system.

With regard to existing classifications of agent systems, interaction agents can employ characteristics that are mostly met in personal assistants and believable agents [5]. A further area of work that is relevant to this paper is related to pedagogical agents [20,34,24,25]. Pedagogical agents employ features that add expressiveness and believability to the look of the user interface, - features that have proven to enhance user motivation and satisfaction [25]. As one might expect, there is keen interest from the Human Computer Interaction (HCI) community in all types of interface agents, although there is little work [39] known to the authors which specifically discusses the design of such agents.

This paper describes how HCI techniques such as task analysis [18,19,4], and usability engineering [29,30,7,3] can be used to define and design this type of agent, by examining the whole of the interaction space and defining which parts should be associated to the interaction agent. More specifically, this paper presents the identification, design and implementation of a user interface for an on-line brokerage system and the conceptual architecture and functional behaviour of an intelligent interaction agent that supports and enhances the interaction between the user and the system.

This work is being carried out in the context of a European Union ACTS (AC 221) project GAIA. [1,2] The GAIA project is developing a sector and supplier independent Generic Architecture for Information Availability based on brokers, to support multilateral information trading. GAIA offers a robust platform for electronic commerce based on the concept of brokers. A broker integrates a variety of electronic commerce functions and components, namely: authentication, directory services, search, order, item and stream delivery, payment, tariffing, alerting [1,2]. The GAIA system is distributed, based on CORBA [16], and is being demonstrated in three application domains: music, technical components and publishing.

In the next section, the problems that information brokerage systems face in terms of usability are presented, and a solution based on interaction agents is proposed. Section 3 discusses the GAIA user interface designed and developed for GAIA, in terms of the knowledge identified for performing tasks, the technical approach taken and usability issues and development. Section 4 concentrates on the design of the interaction agent, describing its conceptual architecture, inference mechanisms, and functional architecture. Section 5 concludes with summary and discussion of future work.

2. Information brokerage systems - usability problems

Currently, information brokerage systems are capable of searching enormous quantities of data, often from heterogeneous information sources. In this they can be extremely efficient, producing wide-ranging result sets, drawn from repositories sited globally. In addition, they provide links into supplier sites, and a whole host of value-added information. However, electronic brokers need to provide some means of personalising customer services to be effective at channelling information from suppliers to customers. By analogy, human brokers, such as travel agents, or librarians, before commencing a search, spend time in discussion with their clients to better ascertain their needs. After a broker collects information for a client, he spends time discussing the significance of the information with the client, weighing it up and assessing whether or not it is the information required by the client.

This mediation phase is non-existent or very rudimentary in most information brokerage systems. They rely on the input to the system being a fairly precise expression of the customer's requirements. This gap between information availability and the relevance of information to the customer often results in "information overload" in various guises. This can be seen, for instance, in the sheer overwhelming volume of returned results, in information being returned that is not relevant, or in some of the material not being accessible to the user because he does not possess the necessary computer infrastructure or is not sufficiently expert in computer literacy to be able to "decode" various formats, etc. The result is frustration and exasperation, and loss of precious time.

At the same time, at the other end of the supply chain, information providers know that the quantity of information they provide is increasing and that unless their customers are provided with useful and usable ways of accessing that information, customers will go elsewhere.

Further complexity is introduced by the provision of novel user interfaces that integrate a variety of services, from searching and retrieving information to on-line ordering and delivering of digital goods, incorporating tariffing and payment facilities. Nor is complexity restricted to these types of services. As bandwidth expands, and new networking technologies progress, the information that is transmitted is increasingly multimedia based. In addition, the possibility to link up devices such

as cellular phones further increases the functionality and complexity of the interaction space.

Information system developers attempt to tackle the problems of both user and information providers and combat the negative effects of information overload and user interface comprehension and usage with well known solutions of search engines, personal agents, and other software that are offered commercially. However these are often piecemeal solutions which offer limited help. Even after using them, users may still have a vast result sets returned and information providers still cannot be sure that customers will access their information when it's hidden in huge result sets.

In this paper interaction agents are proposed as a metaphor of the agent/personal assistant, for computer–based applications that are novel and possibly difficult to use (at least for non expert users) and which tend by their very nature to produce information overload. Thus the interaction agent functions as both an assistant of the computer-based system and (following the metaphor of the human assistant) of the application domain. This paper focuses on a generic architectural framework based on HCI principles and known AI cognitive architectures that helps the designer to identify the interaction agent design. In the focus of the paper, this architectural framework makes use of user requirements, provides the interaction agent conceptual architecture, and helps to clarify its functional behaviour.

3. GAIA User Interface Design and Development

The design of the GAIA user interface has been based upon user and domain requirements [35], commonly accepted usability principles, metaphors and paradigms [8], as well as on task analysis which was performed upon users and existing electronic commerce user interfaces. This section presents a brief view of the task analysis performed and briefly demonstrates the GAIA user interface. Task analysis identified the knowledge an entity (user or artificial agent) would require to achieve its goals, the stages of interaction at which a user would require interaction agent assistance, and alternative plans that this entity would have to execute in order to perform a (sub) task. The purpose of presenting the GAIA user interface is to show how a user interface design that is based on human-computer interaction techniques can be employed to identify the parts of user-system interaction that can cause difficulties to users for various reasons. In this way, it can provide the justification of the interaction agent as well as input for the design of the conceptual and functional architecture of the agent application. While the domain of application presented is a brokerage system, it is considered that an interaction agent can be employed in other application domains that have heavy user interface requirements due to their novelty and the variety of functionality provided.

3.1. Identifying the Knowledge for Performing User Tasks

Among a variety of known and applied techniques [4,8], the Task Knowledge Structures (TKS) technique [18,19] was selected to identify the knowledge for the performance of user and agent tasks. TKS identifies the knowledge about various components for a given task: roles, goals, sub-goals, sub-tasks, procedures, strategies, actions and objects. It can assist the analyst to capture interaction requirements by identifying the stages of interaction and the knowledge required in each stage of it, in order to perform a task. It does not, however, describe the particular form of presentation that the user interface might take.

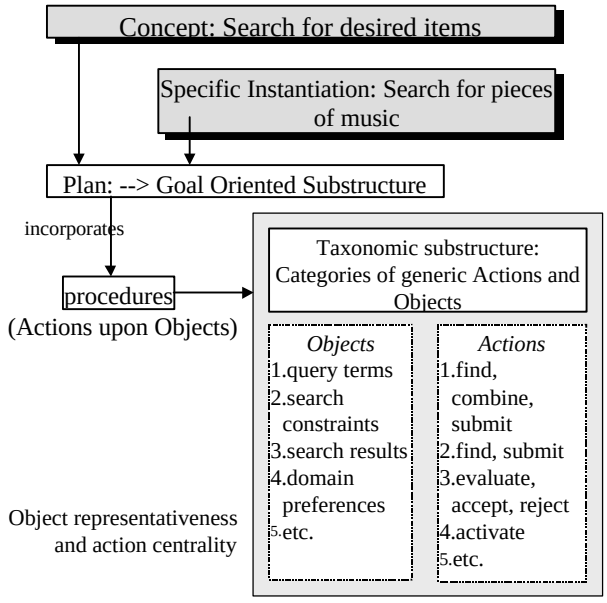

Figure 1 "Search for Desired Items" TKS.

Figure 1 provides an approximation to describe a user's TKS when the task of 'searching for desired items' is carried out. The actions the user needs to perform in this TKS can be separated into *actions that may put heavy cognitive load on users* (find, evaluate, activate), and

actions that are user interface related: actual instructions to the system that may also cause difficulties to users unfamiliar with the interface.

Thus, the interaction agent can assist the user in both contexts. In the case of actions that require domain knowledge, the interaction agent, as an entity knowledgeable about the domain, can suggest to the user information items that may be of interest or query terms to issue a search request. In the case of user actions that are performed on the user interface, the agent can guide the user in manipulating the application user interface.

TKS results in sets of plan-goal oriented substructures (plans) that the users would need to follow in order to perform the task. An example is shown in figure 2.

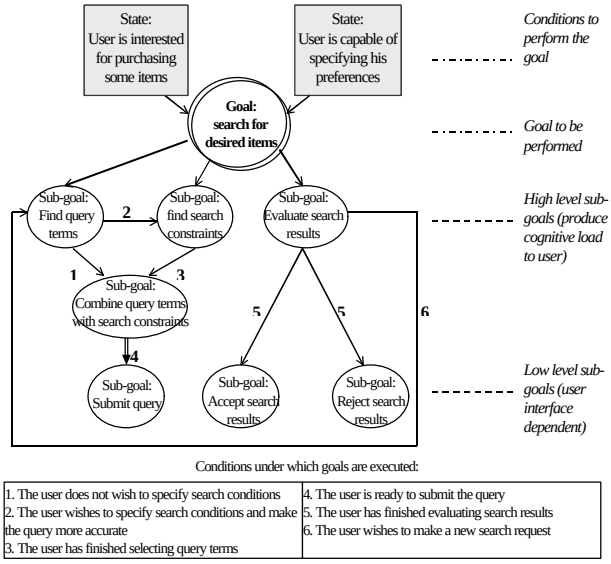

Figure 2 A "Search for Desired Items" Plan.

The production of such plans is very important, as this is a basic feature of the interaction agent architecture. The interaction agent can evaluate user behaviour (as this is interpreted by user actions upon the application user interface) against the set of plans provided by TKS, in a manner that will enable the agent to reason about user behaviour and adjust its behaviour accordingly.

The application of TKS results in a definition of user tasks, alternative plans for the performance of those tasks, and the identification of user-system interaction stages. Furthermore it reveals stages of interaction where actions and objects used typically place cognitive load on the user in the contexts of employment of domain knowledge and actual manipulation of the system. The purpose of the interaction agent is to provide assistance to the user in both contexts.

3.2. GAIA User Interface

3.2.1. Technical Approach. The design and development of the GAIA user interface needed to take into account: the findings of the task analysis performed; generally accepted usability principles and metaphors in HCI design; the rich functionality of the GAIA brokerage system. In addition, GAIA should be widely accessible (i.e. Internet-based), and the user interface tool to be used should offer a variety of capabilities to the development team. More specifically the technical requirements that had to be met by the user interface technology to be employed for user interface development, were: support for CORBA calls to the GAIA system, support for audio and video streams; provision of a rich user interface component set. All the above contributed towards the selection of Java for the development of the GAIA user interface. In particular, the GAIA user interface is implemented in version 1.1.5 of JDK (Java Developer Kit) and uses version 0.6.1 of Swing (user interface) components. The remainder of this section gives an overview of the implementation of the GAIA user interface (configured for the technical components domain) while it discusses the usability issues and the justification for providing an interaction agent as a user assistant for the interaction reflected by this user interface.

3.2.2. Usability Issues and Development. The design of the GAIA user interface paid regard to usability principles and guidelines highlighted in human–computer interaction literature, such as consistency, robustness, predictability, as well as user avoidance of error and information overload [8,26].

As revealed from user requirements [35] the working tasks a user needs to perform are typically three: search, evaluation of search results and order. The GAIA interface has therefore been designed so that these three basic user tasks that can be performed from the three subspaces into which the user interface is divided (figure 3). The subspaces cannot be activated simultaneously. In order for the search results subspace to be activated, the user has to perform a search request. The same happens with the activation of the ordering subspace.

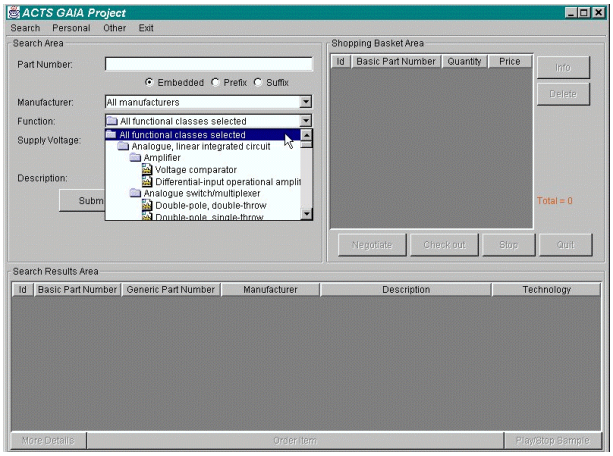

Figure 3: The user works in three user interface subspaces

These constraints contribute towards three aims: a) to making the interface consistent in terms of the way and space in which a (sub)task will be performed, since all options with regard to a (sub)task are available in its corresponding subspace, b) to making it difficult for the user to make an error, because other options are not activated and c) to guiding in this way the user during interaction. Consistency is achieved with regard to another feature as well: tasks that compose a task category are performed in the same way.

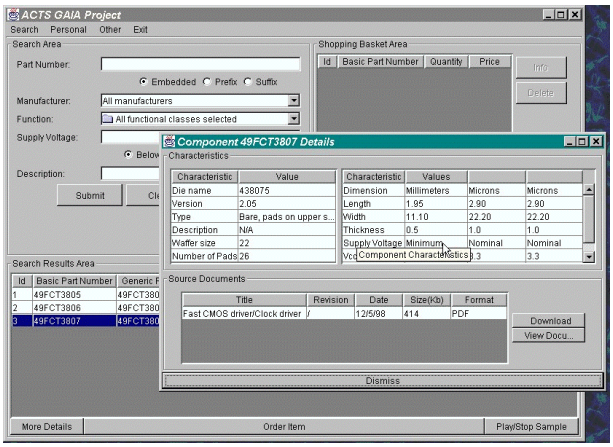

Figure 4: the user sees search results in detail

Additional information is displayed using pop up windows. For instance when the user wishes to be presented with more information about a search or order item (figure 4). This happens in order to avoid information overload. However the user is not presented with more than one pop up window at a time (as often seen in Windows based operating systems: Win95 and NT, as well as in other commercial applications). Apart from reducing cognitive load, this also means that a user can access a user interface function by performing at the most two logical steps.

With regard to robustness, the user is provided with clear exits or cancel points, in every action where this is required. Again, the history of information about the latest user actions, as well as the storage of past sessions' shopping baskets is provided.

The use of well known paradigms of real life metaphors is of great importance, since it contributes towards learnability and specifically, predictability. This interface employs the use of tables, which is a user interface component commonly used in tasks that are relative to management of information and has been used in logistics applications, email management (Eudora), etc. The shopping basket metaphor, used already by many shopping interface applications on the Web, has been also employed in this interface.

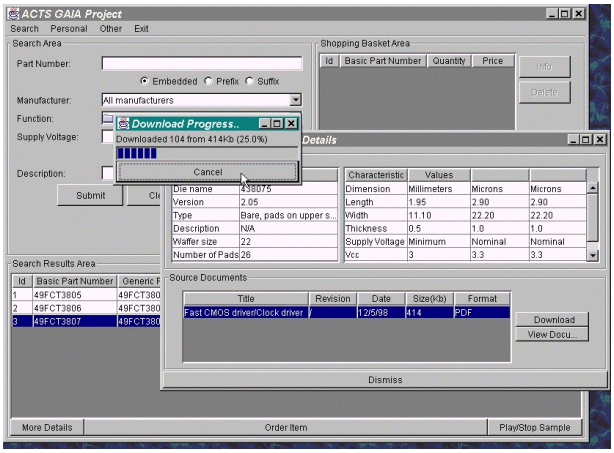

Figure 5: The user can view or download description files

Users can also download files that describe the presented information (figure 5) and save them in their hard disk, or view them by external viewers. For items that are not provided without a fee, users have to order them. The shopping basket subspace is used for users to store information items, while interacting with the system (figure 6).

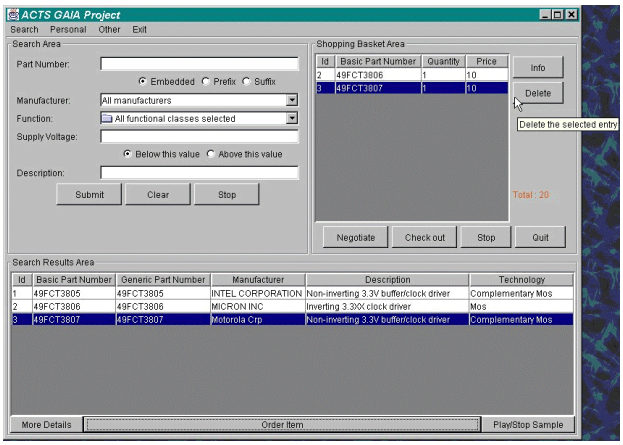

Figure 6: The user puts items into his shopping basket

Video conferencing facilities are also to be incorporated into the user interface in order to support on-line ordering negotiation with information/goods suppliers. Additionally this user interface is being integrated with the user interface component that performs secure payment.

The richness of the functionality of the interface and the interaction space require that some kind of interaction agent act as a personal assistant to the user to help to use the interface and make use of the wide range of system functionality. The manner by which the interaction agent functions can be shown by its architecture. The next section presents the interaction agent architectural design, describing interaction agent architectural components and presenting a set of scenarios with regard to the interaction agent behaviour.

4. Interaction agent architectural design

4.1. Interaction agent conceptual architecture

The interaction agent observes user actions over the application user interface and is capable of reasoning about these actions and deciding whether the user is in a state where he would require the interaction agent assistance. The manner by which the interaction agent reasons about its own and user actions, is determined by its architecture.

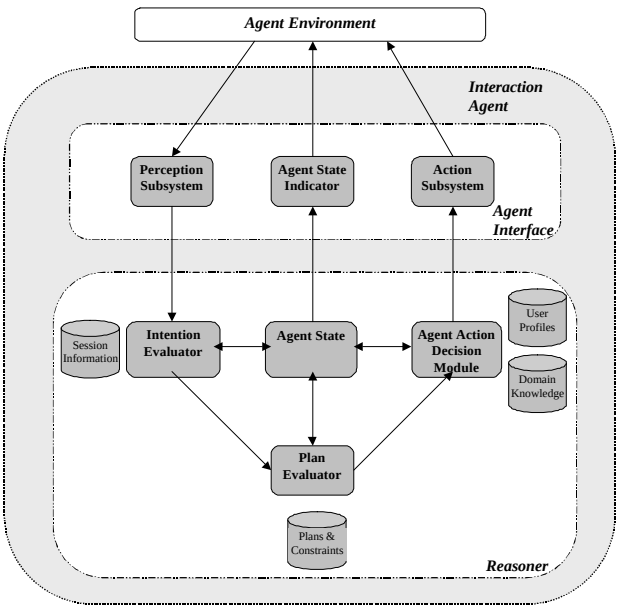

Figure 7: Interaction Agent Conceptual Architecture

The interaction agent architecture is based on the findings (tasks and alternative plans for performing each task) of the TKS application. This architecture has been influenced mainly by three cognitive architectures: SOAR [22], the Opportunistic Control Model [13,14], and O-Plan Knowledge Framework [37]. It is represented graphically in figure 7 and its components are described below.

The interaction agent continuously observes actions occurring upon its environment via its **perception subsystem**. The **agent environment** in this case consists of the application user interface, as well as the underlying brokerage system. Among all the possible actions that the user can perform upon the application user interface, there are actions (and especially results of user actions) that will stimulate the agent and may result in agent initiatives. The **intention evaluator** identifies such user actions and may either affect the **agent state** or further evaluate the user plan by invoking the **plan evaluator**. The intention evaluator needs to keep session information in order to have a continuous view of user actions. The outcome of the intention evaluator is a (set of) user state condition(s). Such states have been identified by TKS and actually indicate high-level user goals; example states include: “the user is (not) capable of performing this task”, “the user has a specific query from the system”, “the user may not be able to express his query to the system”, etc. Such estimations of user

states, as well as the actions the user performs on the user interface are necessary input for the plan evaluator, as these are the actual constraints on the world of plans for the agent (this holds for plans designed according to TKS).

The main purpose of the agent state module is to continuously notify the user about the agent state. In the context of this specific application, the agent state does not affect the agent behaviour drastically, nor does the interaction agent exhibit character behaviour (e.g. selfish, honest, etc.) and thus function differently with regard to its state or character. Nevertheless the interaction agent may (decide or be instructed to) function in a 'guide of the application interface' or 'advisor' mode, which will eventually affect both its set of states and behaviour.

The **plan evaluator** evaluates user actions upon the set of possible plans provided by TKS analysis. The plan evaluator's function results in a full fulfilment of a plan, or a partial or non-fulfilment. In the first case (fulfilment) the interaction agent assumes that the user is capable of using the system and just continues monitoring his actions. In the second case (partial or non- fulfilment) the **agent action decision module** is activated. This results in initiating the interaction agent in either of its two basic modes: that of providing assistance to the user in order to manipulate the application user interface, or suggesting information items. User profiles and domain knowledge are required in this case in order for the agent to provide personalised assistance to each user. The **action subsystem** will provide such assistance on the user interface level of the interaction agent.

4.1.1 Inference Mechanisms Employed. The selection of inference mechanisms employed in interaction agent design is examined separately for each task the agent needs to perform, and with respect to the interaction agent architecture. With regard to the agent tasks of assisting user to search and evaluate search results, collaborative filtering algorithms [6,12,31,33] are implemented to allow the interaction agent to propose information items to users. There is a variety of implementations of collaborative filtering algorithms in the agent literature [31]. Collaborative information filtering approaches recommend information items that other users with similar preferences have rated highly in the past and are highly efficient [6,9,12,33] in a variety of perspectives. For example, information items need not to be well structured (meta-data representation), secondly users are presented with items that may be totally new to what they had in mind. This can be a very positive point for a brokerage system. Then from the profile building aspect, there is no need for eliciting user feedback, it is enough to just to track down meaningful user actions. Nor is there a need for a sophisticated user profile structure explicitly provided by users, their purchases over the system constitutes a source of information, which is adequate for the collaborative filtering algorithm. This has the added advantage that it could also resolve potential problems of user reluctance to use the system.

With regard to assisting the user during ordering, a rule–based inference mechanism is implemented which takes into account both quantitative (domain independent) and qualitative (domain dependent) ordering attributes.

4.2. Interaction agent functional behaviour

This section presents a set of example scenarios of user sessions, within the GAIA system, when the interaction agent is activated. The purpose of these scenarios is to further explain the interaction agent conceptual architecture and the tasks that each component performs within this architecture. After authentication, the first task that the user will (according to TKS) perform is to issue a search request. In the following table some differing user reactions to the user interface and the corresponding behaviour of the agent components and actions are laid out.

Table 1: Scenarios of differing user reactions and corresponding behaviour of the agent components when performing the task of 'searching for desired items'.

PS = Perception Subsystem; IE = Intention Evaluator; PE = Plan Evaluator; AADM = Agent Action Decision Module; ACS = Action Subsystem; AGS = Agent State
Scenario 1: PS: User presses submit button without specifying search criteria. **IE**: The user may not be capable of using the system. **PE:** No plans are associated with such an action. **AADM:** Agent will suggest assisting the user in order to use the system. **ACS:** Agent guides the user to issue a search request **AGS:** Suggesting user interface assistance.
Scenario 2: PS: User issues a search request by filling attributes that are unique or fairly specific to an information item. E.g. ISBN in the publishing domain, generic part number for the technical components domain.**IE**: The user has a fairly specific request (might not need interaction agent assistance). **PE:** Agent plans are associated with such an action. Decision module is presented with alternatives. **AADM:** A search request will be made to the system **ACS:** A search request is made to the system. Search results are presented to user.**AGS:** Searching, Presenting.

Scenario 3: PS: User issues a search request by filling attributes that are not unique or fairly specific to an information item. E.g. subject in the publishing domain, description for the technical components domain. **IE**: The user does not have a specific request for the system (might need interaction agent assistance). **PE:** Agent plans are associated with such an action. Decision module is presented with alternatives. **AADM:** A search request will be made to the system**. ACS:** A search request is made to the system. Search results are presented to user. **AGS:** Searching, Presenting.
Scenario 4: PS: User explicitly requests from the interaction agent to suggest information items that may be of interests**. IE**: The user may not have a specific request for the system and requests interaction agent assistance explicitly. **PE:** Agent plans are associated with such an action. Decision module is presented with alternatives. **AADM:** Agent identifies items that may be of interest to the user (user profile, knowledge base). **ACS:** Items identified are presented to user. **AGS:** Suggesting.

As shown above, the alternative behaviours that the interaction agent would express to the user would be either:

- to intervene to assist the user to comprehend and manipulate the user interface (when the user performs a user interface action that does not conform with any task plan),
- not to intervene (when user actions are considered by the agent as logical –according to its plans-),
- or to intervene to provide assistance related to domain knowledge (in this case the user asks this explicitly; this could happen in other cases as shown below).

The next task the user will have to perform after a search request is the evaluation/examination of search results. During the performance of this task, the user has a variety of options as shown by TKS. The following set of scenarios presents the alternative user actions upon the user interface and the behaviour of the agent components and actions.

Table 2: Scenarios of differing user reactions and corresponding behaviour of the agent components when performing the task of 'evaluation of search results'.

PS = Perception Subsystem; IE = Intention Evaluator; PE = Plan Evaluator; AADM = Agent Action Decision Module; ACS = Action Subsystem; AGS = Agent State
Scenario 1: PS: User puts items in his shopping basket, without examining them in detail. **IE**: The user may not capable of using the (whole functionality of the) system. **PE:** Such a user plan indicates either thorough comprehension of search results, or inadequate knowledge of the system functionality. **AADM:** The agent will ask the user whether he wants to see search results in detail. **ACS:** Prompting user to examine search results in detail. **AGS:** Querying / Advising.
Scenario 2: PS: User examines search results in detail (e.g. sees more description attributes, downloads audio / video streams). **IE**: The user has a fairly specific knowledge of the system. **PE:** Such a user plan indicates that the user has a good knowledge of the system. **AADM:** The agent will not respond to user actions. **ACS:** No action. **AGS:** Observing.
Scenario 3: PS: User issues a new search request. His previous search request was not specific enough and search results set was irrelevant/too large. **IE**: The user does not have a specific request from the system, while examination of search results is a difficult task. **PE:** Such a user plan indicates that the issued query had no practical results for the user. **AADM:** The agent will propose to refine search results according to user preferences. **ACS:** Present a manageable number of results according to user preferences. **AGS:** Evaluating / refining.
Scenario 4: PS: The system returns no results on a search request that was not fairly specific. **IE**: The user may not be able to express his query to the system. **PE:** Such a user plan indicates that the issued query had no practical results for the user. **AADM:** The agent will suggest information items to the user. **ACS:** Present a number of novel information items to user. **AGS:** Suggesting.

The manner in which the agent intervenes, while the user performs tasks is a very important issue. The user should always have the option to bypass the agent or to specify the level of intervention upon specific agent actions. This notion of control is paramount in user acceptance of the agent. It is always possible that there is no match between the agent plan and intention evaluator. In this case intervention by the agent may be more detrimental than helpful. This happens in human–human interaction as well, e.g. salesmen proposals may frustrate customers, while bypassing salesmen is sometimes difficult as well. For example in the case of the first scenario, the user may indeed have a good

knowledge of his actions and the system may have corresponded to his query as exactly he would expect, thus the action of putting items in his shopping basket without examining them in detail could be reasonable. Such user behaviour is naturally interesting to the interaction agent and it updates the user profile accordingly.

5. Summary and Future Work

The first objective of any widely targeted application is to be usable. Unless users find the manipulation of the user interface straightforward, they won't use it, especially on such a wide information market as the Internet. However, the provision of a robust design and implementation of a user interface may not ensure that users comprehend and readily use an interface. Nowhere is this more apparent than in the case of a user interface of a novel system, such as a brokerage system, providing a wide variety of user (sub)tasks and rich functionality, not to mention committing users to spend money. For users to efficiently use such an interface, the employment of an interaction agent is proposed. The interaction agent, validated from task analysis, can assist the user in two broad contexts:

At the level of manipulating/learning the user interface, the interaction agent can infer about users' ability to use the system and provide assistance to users who seem to be experiencing difficulties using this user interface. Such cases could be when users perform actions that cannot possibly have a logical meaning (i.e. according to the agent model of plans about user actions), when users make errors, possibly when they don't use the whole of the functionality of the system.

At the interaction level, the agent can suggest information items to users and in general suggest options and selections that are of interest to users when the agent reasons that users would need such suggestions. For example, when users do not seem to be able to express their requirements to the system, or when the system's responses to their requests cannot possibly be handled by them alone and need some kind of sorting and evaluation process. More specifically, during searching the agent may suggest new information items to users; during evaluation of search results the agent may offer to refine those results on behalf of the user; during ordering the agent may suggest alternative selections that may be of interest to the user.

In each of these cases it is essential users can adjust the level of agent intervention, and furthermore that users can bypass the agent (or activate it) at any stage of user system interaction.

This paper presented the identification, design and implementation of a user interface to a brokerage system and the conceptual architecture and functional behaviour of an intelligent interaction agent that supports and enhances the interaction between the user and the system. Interaction agents were proposed as a metaphor of the agent–personal assistant, to computer–based applications that are novel and possibly difficult to use (at least for non expert users) which tend by their very nature to produce information overload. While the domain of application of the interaction agent presented in this paper is a brokerage system, it is considered that the work described in this paper can be employed in other application domains that have heavy user interface requirements due to their novelty and the variety of functionality provided.

This work is to be further extended in two directions: the inference mechanisms employed for recommendation of information items and the agent user interface expressiveness. With regard to the first issue this work will be extended in order to employ content–based filtering algorithms. It has been shown [31] that collaborative information filtering and content–based filtering techniques are complementary and can give highly efficient recommendations. With regard to further work on the agent user interface, apart from the employment of a persona and the use of user interface media such as gazing and human voice, there are plans to extend this work with voice recognition.

6. References

[1] ACTS (AC221) GAIA, Generic Architecture for Information Availability: http://syspace.co.uk/GAIA.

[2] Bessonov M., Hands J., Patel A., Smith R., An Inclusive and Extensible Architecture for Electronic Brokerage", submitted to HICSS-32, 1999.

[3] Byerley, P.F. Barnard, P.J. and May, J., editors (1993) Computers, Communication and Usability: Design issues, research and methods for integrated services. (North Holland Series in Telecommunication) Elsevier: Amsterdam.

[4] Buckingham Shum, S Jorgensen, A. Hammond, N. and Aboulafia, A. (eds.) (1993) Amodeus HCI Modelling and Design Approaches: Executive Summaries and Worked Examples, http://www.mrc-cbu.cam.ac.uk/amodeus/.

[5] Cesta A., Collia M., D'Aloisi D., (1998) An interactive agent-based meeting scheduler, First International Workshop on Interaction Agents, IA'98, L'Aquila, Italy.

[6] Communications of the ACM (1997), Special Issue : Recommender Systems, Vol. 40, Number 3, March 97.

[7] Diaper D., (1997) HCI and Requirements Engineering - Integrating HCI and Software Engineering Requirements Analysis, SIGHCI Bulletin Vol. 29 No 1, January 1997.
[8] Dix A., Finlay J., Abowd G., Beale R., (1993) Human-Computer Interaction, Prentice Hall 1993.
[9] Firefly: http://www.firefly.com/.
[10] Foner L., What's an agent anyway?, (1994) online paper: http://foner.www.media.mit.edu/people/foner/Julia/Julia.html .
[11] Franklin S., Graesser A., (1996) Is it an agent or just a program? A taxonomy for autonomous agents, Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1996.
[12] Goldberg, D.; Oki, B.; Nichols, D.; Terry, D. B. (1992) ``Using Collaborative Filtering to Weave an Information Tapestry," Communications of the ACM, December 1992, Vol 35, No 12, pp. 61-70.
[13] Hayes-Roth B. (1991) An Integrated Architecture for Intelligent Agents SIGART Bulletin 2, 1991, 82-84.
[14] Hayes-Roth B (1993) Opportunistic Control of Action in Intelligent Agents, Knowledge Systems Laboratory, Stanford University, IEEE Transactions on Systems, Man and Cybernetics v. 23, 1993 pp 1575-1587.
[15] Haynes T., Sen S., Arora N., Nadela R., (1996) An automated meeting scheduling system that utilises user preferences.
[16] IONA Technologies (1997) , Orbix 2 Reference Guide, IONA Technologies PCL, March 1997.
[17] Jennings N. R., J. Corera, I. Laresgoiti, E. H. Mamdani, F. Perriolat, P. Skarek and L. Z. Varga, (1996) Using ARCHON to develop real-word DAI applications for electricity transportation management and particle accelerator control, IEEE Expert, December 1996.
[18] Johnson P., Johnson H., Waddington R., Shouls A., (1988) Task-Related Knowledge Structures: Analysis, Modelling and Application, People & Computers: from research to implementation, Cambridge Press pp. 35-62, 1988.
[19]Johnson P., Johnson H., (1988) Task Know-ledge Structures: Psychological basis and integration into systems design, Acta Psychologica, 78, pp. 3-26.
[20] Johnson W. L., Learning from agents (1998) First International Workshop on Interaction Agents, IA'98, L'Aquila, Italy.
[21] Koutsabasis P. Darzentas J.S. Spyrou T. Darzentas J. (1998) An interaction agent in a brokerage environment, First International Workshop on Interaction Agents, (IA'98), L'Aquila, Italy.
[22] Laird J., Hucka M., Huffman S., (1991) An analysis of Soar as an integrated architecture, SIGART Bulletin 2, 85-90, 1991.
[23] Lashkari Y., Metral M., Maes P., (1994) Collaborative Interface Agents, In proceedings of the National Conference on Artificial Intelligence, 1994.
[24] Lester J. C., Stone B. A., (1997) Increasing believability in animated pedagogical characters, First International Conference on Autonomous Agents (Agents '97), California, USA.
[25] Lester J. C., Converse S. A, Kahler S. E., Barlow S. T., Stone B. A., Bhoga R. S. (1997) The personal effect: Affective impact of animated pedagogical agents, First International Conference on Autonomous Agents (Agents '97), California, USA.
[26] Maes P. (1994) Agents that Reduce Work and Information Overload, Communications of the ACM July 1994, Vol. 37, No. 7.
[27] Moukas A., (1996) Amalthea, Information Discovery and Filtering using a Multiagent Evolving Ecosystem, Proceedings of the Conference on Practical Application of Intelligent Agents & Multi-Agent Technology, London, 1996.
[28] Negroponte, N. (1970) The Architecture Machine; Towards a More Human Environment, MIT Press, 1970.
[29] Nielsen J., (1992) The Usability Engineering Life Circle, IEEE Computer, March 1992
Nielsen J., (1992) Usability Engineering, Academic Press, 1992.
[31] Oard D. W., (1996) A conceptual framework for text filtering, Technical Report, University of Maryland, College Park.
[32] Petrie, J. C. (1995) Agent-Based Engineering, the Web and Intelligence, IEEE Expert - Intelligent Systems and their Applications, Vol. 11, Issue 6, December 1996, pp. 24-29.
[33] Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., and Riedl, J.GroupLens: An open architecture for collaborative filtering of netnews. Proceedings of the 1994 Computer Supported Collaborative Work Conference.(1994).
[34] Rickel J., Johnson W. L., (1997) Integrating pedagogical capabilities in a virtual environment agent, First International Conference on Autonomous Agents (Agents '97), California, USA.
[35] Smith R., Maroulis D., (1996) GAIA Domain and User Requirements, Deliverable 0301, http://www.syspace.co.uk/GAIA.
[36] Sullivan J. W., Tyler S., W., (Editors) Intelligent User Interfaces, ACM Press Frontier Series, 1991.
[37] Tate A., (1995) O-Plan Knowledge Source Framework, Artificial Intelligence Applications Institute, University of Edinburgh, Technical Report, March 1995, Internet Page: http://www.aiai.ed.ac.uk/~oplan/oplan/oplan-doc.html.
[38] Wang H., Wang C.(1997), A Taxonomy of Intelligent Agents, in Intelligent Agents in Nuclear Industry, IEEE Computer November 1997, pp. 31.
[39] Whatley J. E., Scown P. J. A., (1998) A method to specify interactions between human operators and multi – agent systems, First International Workshop on Interaction Agents, IA'98, L'Aquila, Italy, 1998.