On Intelligent Interface Agents for Human Based Computation*

F. Aznar, M. Sempere, M. Pujol, and R. Rizo

Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
{fidel,mireia,mar,rizo}@dccia.ua.es

Abstract. In this paper a new type of interface agent will be presented. This agent is oriented to model systems for human based computation. This kind of computation, that we consider a logical extension of intelligent agent paradigm, emerges as valid approach for the resolution of complex problems.
Firstly an study of the state of the art of interface agents will be review. Next, human based computation will be defined and we will see how is necessary to extend the current typology of interface agents to model this new kind of computation. In addition, a new type of interface agent, oriented to model this type of computational system, will be presented. Finally, two of the most representative applications of human based computation will be specified using this new typology.

1 Introduction

Knowledge and communications can be considered as two of the most important pillars of our society. Whereas the information of digital systems increases (such as products, services, people, maps, ...), the computing technologies we have develop become, paradoxically, both the gateways to all kinds of resources and the barriers to access them. Therefore, computers become very important in improving our lives and are essential not only to the relatively few of us who have the necessary skills to access to resources, but to everyone [1], [2]. More and more, we must use very specific interfaces to carry out some communication tasks with computers and other electronic devices.

Thinking of this problem, as early as in the decade of the 90, Kay [3] and other authors, being based on the paradigm of intelligent agents, emphasized the importance of an interface agent, that was going to improve the computation as we know it today. An interface agent would allow us to advance from the direct system manipulation to simpler interaction based on agents. This way, the elimination of the necessity of controlling all the interface details allows people to have more time to do another things (people will be able to obtain others goals, that in another way would require an expert).

* This work has been financed by Generalitat Valenciana Project ARVIV/2007/071

Instead of establishing a direct interaction with commands or a direct manipulation of the interface, the user is introduced in a cooperative process where the human agent and the interface agent start a communication, monitoring a set of events and carrying out a group of tasks. An analogy between an interface agent and a personal assistant can be drawn, because this agent is collaborating with the user in the same work environment [4]. If this cooperative process between the user and the interface agent is established in a closer way, even symbiotic, we find a new computational paradigm: human based computation.

Human based computation can be defined as a technique that performs computational problems requesting the resolution of certain steps of the problem to humans. This approach is different from traditional computation, where a human employs a computer to solve a problem. In this computation, a human provides a formalized problem description to a computer and receives a solution to interpret it. In human based computation, the roles are often reversed: the computer asks a human (or several) to solve a problem, collecting, interpreting and integrating all the solutions.

The importance of this computation, that makes that companies as google (`www.google.com`) includes them in search engines, lies in the use of the innate ability of humans to detect patterns in order to help computers (that are less suitable to this task, although computers are better than humans in other tasks, such as detecting randomness in a process, ...) [5].

In this paper, we will review the state of the art of current interface agents that learn from the user, analysing their characteristics, advantages and disadvantages. Then, a new type of interface agent able to perform human based computation tasks (HC), will be specified. Therefore, we will show both the characteristics that define a human based computation process and the roles and actors that must be present in this type of computation. Although some applications based on HC exist, we have not found any specification or formalization based on intelligent agents, being human based computation, from our point of view, a logical extension of interface agents. Some of the most representative applications based on human based computation will be analysed from the point of view of interface agents. Finally conclusions and future work will be drawn.

2 Intelligent Interface Agent Systems

An interface agent can be defined as an intelligent system which assists users with daily computer based tasks [6]. This agent allows the user to delegate difficult tasks with a main purpose, to reduce the workload of users by creating personalized agents to which personal work can be delegated.

From the point of view of agents that learn from the user, three types of interface agents exist: user programmed agents, where the user must program the agent; agents that receive user feedback, where the interface agent is endowed with an extensive domain model and a user model, and agents that monitor user

behaviour, which are provided with the ability to program theirself, for example, acquiring the knowledge that they need to assist the user [4].

When designing this type of agents it is necessary to consider two main criteria: the competence of the agents, that is, it should be specified how the agent acquires the knowledge that needs to decide when and how to help the user; and their trust, we should know how we can guarantee that the user feels comfortable delegating the tasks to the agent.

The user programmed interface agents are the simplest to implement but they are not always the most appropriate, because their operation not always denotes intelligence. In this type of agents, end users are who directly provide the rules and the criteria of operation of the agent. Their main problem is that their operation depends directly on the abilities of the user to program the agent.

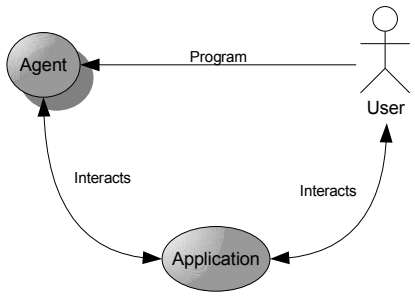

Fig. 1. Interaction between actors in an user agent programmed

The operation of this type of agents is summarized in the figure 1 where the main flows of information and the actors that are part of the environment are shown. As can be observed, an agent is programmed by the user to help him with the interaction with the application.

These agents do not satisfy the competence criterion since they require a lot of knowledge and user effort. In addition, the user has to recognize the opportunity to use the agent, to take the initiative to create it, to provide the necessary knowledge as well as to maintain it consistent in time. Regards trust criterion, this type of agents are not problematic because the user is who program them.

Agents that receive user feedback. These type of agents are usually created using knowledge based systems. These agents could receive explicit feedback or only an initial training. Their development is quite complex but they usually show intelligent behaviours.

In figure 2 can be observed the operation of this type of agents. A knowledge engineer must program the agent in a way that it accepts the knowledge of the user. The agent will use this knowledge to interact with the application and to improve the interaction of the user with the application.

Regards the competence, these agents need a lot of knowledge to be embed into the agent. Sometimes, this agent cannot be reused in other applications.

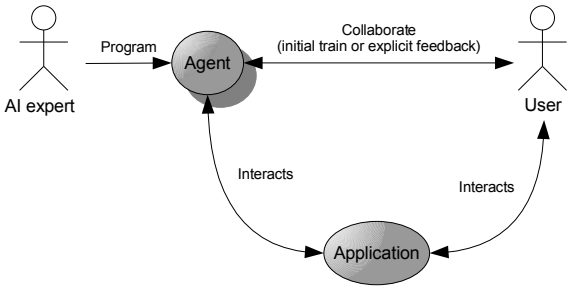

Fig. 2. Interaction between actors in an agent that receive user feedback

On one hand, in the case that the agent only receives an initial training, the knowledge of the agent is prefixed so cannot be adapted to habits or preferences of the user. On the other hand, if the agent is programmed by an external person the trust is reduced because, for example, the agent could have programmed a model that the user does not know.

Agents that monitor user behaviour extract patterns from the actions of the user or even of other users. The complexity of developing these agents is middle and its operation should be correct in specific areas. An agent of this type needs some knowledge of the environment to learn a *correct* behaviour from the user or other users.

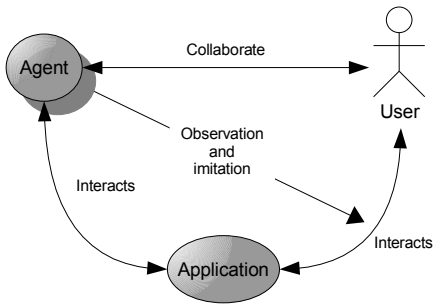

Fig. 3. Interaction between actors in an agent that monitor the user

The operation of this type of agents can be observed in the figure 3. Both the agent and the user interacts with the application. Nevertheless the fundamental difference with previous approaches is that the agent monitors the interaction between the user and the application to learn from it. This type of agents acquires its competences from four sources of information:

– Monitoring the user while the user is interacting with the application
– With direct or indirect feedback with the user
– From examples provided by the user
– Asking to other agents that assist other users and have the same task

Agents that monitor user behaviour will be more and more competent as they learn the user preferences. Regarding the trust, these agents will develop their abilities gradually. The agent will also be able to give explanations of its reasoning and behaviour in a language close to the user.

3 Interface Agents for Human Based Computation

As we commented above, human based computation can be defined as a technique that performs computational problems requesting the resolution of certain steps of the problem to humans. This technique is being used for years and now it is very important due to the growth and development of information networks. Next, we will present a new type of interface agents oriented to perform this type of computation.

If we base on the basic characteristics of a flexible system, it is easy to realize that an intelligent agent perfectly adapts to this kind of computation. On one hand, it must be a social agent, that is, it must interact in an appropriate way with other agents and humans so that it must be able to complete its problem helping others with their tasks. On the other hand, it must be pro-active and therefore, it must act in two ways: answer to the environment and be able to look for the opportunity and to take the initiative when it considers appropriate.

In the classification of interface agents presented in the previous section the agent that is adapted better to the human based computation is the agent that monitors user behaviour. Nevertheless there are some characteristics that are not captured in this type of agents and are necessary for a human based system:

New sources of information. In agents that monitor user behaviour the information is obtained from the interaction of the user. In human based systems, the agent can request to the user or to other agents some help to solve a computational problem.

Deeper interaction between agents and the user. A new communication more deeper than the feedback between agents and humans exist. In contrast to classic interface agents where the interface agent is considered as an assistant of a human, in this case, both human and artificial agents must collaborate in the same level to solve a certain problem.

Completely different agents can collaborate to each other to perform this type of computation. Collaborative systems of interface agents are usually based on the exchange of information between agents with the same characteristics that interact with different users [6]. In human based computation can exists different agents that are able to exchange information obtained from the users to solve a certain problem.

This way, we see the necessity to specify a new type of interface agent oriented to human based computation.

3.1 Agents for Human Based Computation

An agent system for human based computation can be defined as a group of intelligent agents that are organized as is shown in figure 4. These agents must have the next characteristics:

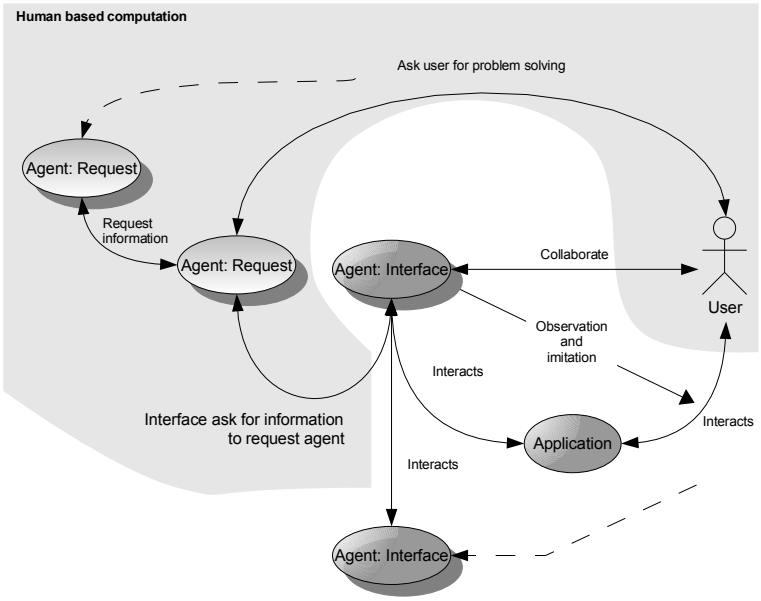

Fig. 4. Interaction between actors in an agent based on human computation

Request agents. There are some agents that request information to the user. These agents form the basic process of human based computation. They set out computational processes to human so that humans must solve them collaborating with the system. Request agents can request information to other agents with the same role, this way, an agent can obtain information from several users.

Request role is not exclusive and an interface agent can develop this role, so that, the minimum system based on human computation could have only one agent that develop the request role.

Any agent that communicates with the user can provide the request role and therefore it can provide valuable information about computational problems that it has delegated to the user. The agents that act as request could be very different from each other (in fact, the more different, more information they will obtain from the user).

Resolution of computational problems. Request agents can ask for the intervention of the user in different ways, always to solve a computational

problem that they cannot deal with. It can be an explicit request or also it should be common to develop an agent for a different application (normally playful) designed specifically to obtain the data needed from the user (a lot of times the user is not conscious of this).

Interface agents. Interface agents act in the same way than the agents that monitor user behaviour but these agents can obtain information from a new source asking to request agents.

It is important to highlight that the information used by request agents is not the same that the information of other interface agents. As we commented above, interface agents have the same design for a certain application and therefore their behaviour is very similar. In contrast, request agents can be very different and therefore they can provide more information than a group of interface agents.

The fundamental difference between the roles of interface agents and request agents is that interface agents make predictions, comparing new situations with older situations and they try to obtain confidence measures of the system operation, for example, asking to other interface agents. In contrast, the main purpose of request agents is to ask for the solution of a computational problem and obtain and process the result of this problem.

Interface agents systems based on human computation, like agents that monitor user behaviour, are gradually being more competent. Moreover, the learning process is faster due to the fact that they can ask for more specific information to request agents. In the same way, an agent will develop its abilities whereas the user learns the agent operation. In this case, the agent has been designed using data obtained from human computation, so its operation is closer to the user, since the agent can even give explanations extracted directly from human reasoning.

The main advantages of the use of agents based on human computation can be summarized in the following points:

- In human computation, each agent try to solve those problems that it is able to face up to in a better way, simplifying the resolution as well as improving the obtained results.
- The results obtained by a lot of users are fused in order to obtain a general conclusion that can be used for other problems or even as a base for a general resolution model. In this paper we will not discuss the advantages of information fusion, that can be consulted in [7], [8], [9].
- Many applications of human computation take advantage of the free time of humans in order to perform useful operations, using this time of calculation that a priori it is wasted.

4 Reviewing Human Based Computation Applications

In this section two of the most common applications of human based computation will be analysed from the point of view of intelligent agents. This way we will see

how these applications can be formalized using the new type of interface agents proposed.

4.1 Google Image Labeler

Several of the most important applications related to the field of human based computation arise from the team directed by Luis von Ahn. For example, Phetch [10] is an on-line game that tries to improve the accessibility to the Web. Players have to label a set of images that will be used to provide alternative texts. This is very important to provide access to visual impaired people. Peekaboom [11] is other collaborative game where a player tries to guess the image that the opponent is hiding. The opponent only shows some parts of the image. Data obtained by this game are used to train an artificial vision system.

In this section we will model the main interactions of the application Google Image Labeler (`http://images.google.com/imagelabeler`). In this application two people must be in agreement to assign tags, labels and keywords to random images that come from Google Images. The more images you guess, the more points you get. The result will be quite good if two people were able to be in agreement, taking into account the difficulty of this problem for a computer. This application was of a project of the research group directed by Luis von Ahn and was licensed to Google Inc.

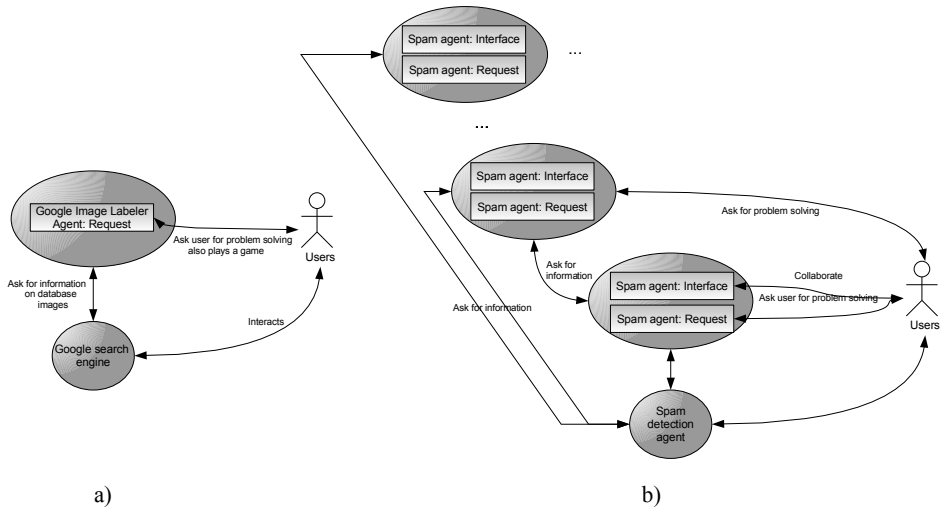

Fig. 5. a) Viewing Google image labeler as an agent system, b) Spam filter system modeled with interface agents

As can be seen in the figure 5 Google Image Labeler is a playful application that implements a request agent. This agent obtains information that will be

used in google search engine. This engine could use different interface agents that interact with users, but by default any agent of this characteristic is provided so the user interacts directly with the application.

Google Image Labeler is one of the most representative applications of human based computation, that is not modelled with agent paradigm. Nevertheless, it is a usual structure of applications for human based computation that have not been designed with an agent oriented methodology. An application for human based computation must have al least one request agent interacting with the user and with the main application, as in this case. User experience can be improve integrating an interface agent to this application. On one hand frequent questions of the users could be learned. In addition, other users that use the same searching patterns can be looked for to share results. Some of the improvements proposed can be consulted in [6].

4.2 Spam Detection

Other applications that use human based computation are spam detection applications. In Spam net web site (`http://www.cloudmark.com`), Vipul's Razor web site (`http://sourceforge.net/projects/razor`) and in [12] each mail received by the user is marked as spam with a specific button (if it is the case) by the user. This way when some people mark the mail as spam, this mail will be considered automatically spam for the rest of users.

The figure 5b shows the general interaction diagram of these applications. In this case, some request agents that obtain information from the users and from others agents are defined to provide a global knowledge. In addition, each request agent shares the role of interface agent, and must collaborate with the user. To split both roles in different agents depends on the problem to solve. Sometimes it is relatively easy to create external applications with agents that developing the request role will obtain the needed data for a certain tasks. Whenever it is possible, is a good option, since it can allow to access to a bigger number of users.

5 Conclusions

In this paper a brief state of the art of interface agents has been presented. Based on this review a new type of interface agent, that can model human based computation process, has been specified.

Moreover, human based computation process has been review, specifying its most important features, and analyzing them as a logical extension of the intelligent agents paradigm. Also we have showed two of the most important applications of this kind of computation, providing an agent based design, using the proposed type of interface agent.

We see this work as a necessary basis for elaborating interface agent systems that use human based computation. This kind of systems can use all the advantages of both, agent paradigm and human based computation. Based on

the proposed model we are now developing applications oriented to computer vision and robotics for learning human behaviour patterns that could be useful for computers.

References

1. Brewer, E., Demmer, M., Du, B., Ho, M., Kam, M., Nedevschi, S., Pal, J., Patra, R., Surana, S., Fall, K.: The Case for Technology in Developing Regions. Computer **38**(6) (2005) 25–38
2. Sims, K.: Artificial Evolution for Computer Graphics. SIGGRAPH Comput. Graph. **25**(4) (1991) 319–328
3. Kay, A.: User Interface: A Personal View. Addison-Wesley (1990)
4. Maes, P.: Agents that Reduce Work and Information Overload. Comunications of the ACM **37** (1994)
5. Norving, P.: Mistakes in Experimental Design and Interpretation. http://norvig.com/experiment-design.html (2007)
6. Lashkari, Y., Metral, M., Maes, P.: Collaborative Interface Agents. citeseer.ist.psu.edu/lashkari94collaborative.html **1** (1994)
7. Llinas, J., Bowman, C., Rogova, G., Steinberg, A., Waltz, E., White, F.: Revisiting the Jdl Data Fusion Model II. Proceedings of the Seventh International Conference on Information Fusion (2004) (2004)
8. Waltz, E. & Llinas, J.: Multisensor Data Fusion. Artech House (1990)
9. Brooks, R., S.S., I.: Multi-Sensor Fusion: Fundamentals and Applications. Prentice Hall, New Jersey (1998)
10. von Ahn, L., Ginosar, S., Kedia, M., Liu, R., Blum, M.: Improving Accessibility of the Web with a Computer Game. In: CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems, New York, NY, USA, ACM Press (2006) 79–82
11. von Ahn, L., Liu, R., Blum, M.: Peekaboom: A Game for Locating Objects in Images. In: CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems, New York, NY, USA, ACM Press (2006) 55–64
12. Zhou, F.; Zhuang, L.Z.B.H.L.A.D.K.J.: Aproximate Object Location and Spam Filtering. ACM Middleware (2003)