A Taxonomy of Web Agents

Zhisheng Huang*, Anton Eliëns*,+, Alex van Ballegooij+, and Paul de Bra**,+

*Free University of Amsterdam, The Netherlands
+Center for Mathematics and Computer Science(CWI), The Netherlands
**Eindhoven University of Technology, The Netherlands
{huang,eliens}@cs.vu.nl, alexb@cwi.nl, debra@win.tue.nl

Abstract

In this paper we propose a taxonomy of web agents, which encompasses agents that provide a text-based interface to for example information retrieval services as well as avatar-embodied guides that help visitors to navigate in virtual environments. Our taxonomy must be regarded as an instrument to delineate targets for research and the realization of prototype applications that demonstrate the usefulness of agent-based intelligence on the Web. In addition, we deploy our agent-taxonomy to establish the implications particular target applications have with respect to software architecture and computational resources.

1. Introduction

There is a lot of interest and work in the research and development of agent technology with applications on the Web. Many types of web agents have been proposed in recent years, which range from domain-dependent agents, like e-commerce agents, information gathering agents, to function-dependent agents, like negotiation agents, co-operating agents. In addition, we like to mention the existence of virtual community agents. The natural questions related to that phenomenon are: what are the relations among so many different types of web agents? Are they redundant, or overlapped? Is there any taxonomy to classify them?

In this paper, we will propose a taxonomy of web agents, which encompasses agents that provide a text-based interface to, for example, information retrieval services, as well as avatar-embodied guides that help visitors to navigate in virtual environments. Our taxonomy must be regarded as an instrument to delineate targets for research and the realization of prototype applications that demonstrate the usefulness of agent-based intelligence on the Web. In addition, we deploy our agent-taxonomy to establish the implications that particular target applications have on the software architecture and computational resources. This research is part of a Dutch research project WASP, for Web Agent Support Program. As part of this research project we are developing PAMELA, a Personal Assistant for Maintaining Electronic Archives. The taxonomy we present in this paper is meant to serve in selecting a suitable "type" of agent architecture to use in the WASP project.

This paper is organized as follows: Section 2 gives a brief overview of intelligent services on the Web. Section 3 and Section 4 discuss what role agents can play in virtual environments and games. In Section 5, we propose the taxonomy. Section 6 indicates research issues which need to be addressed to realize the potential of intelligent web-agents. Section 7 concludes the paper.

2. Intelligent services on the web

In a relatively short time, the Web has become a de facto standard for the dissemination and retrieval of information. Due to the exponential growth of the Web and the information it provides, finding relevant information has become more and more difficult. In particular, browsing is in most cases no longer appropriate for the user searching for specific information. It is our view that, in the near future, access to the Web will increasingly be mediated by intelligent helper applications, software agents, that assist the user in finding relevant and interesting information. As testified by a large body of literature, including [2, 3, 8, 7], this view is shared by many authors and developers.

As a working definition of the notion of *agents*, we adopt the characterization given in [6], which includes characteristics such as *responsiveness*, *pro-activity* and *socialness*. These characteristics, obviously, are strongly metaphorical, and are meant to stress the need for flexible, adaptive approaches to provide assistance to the user.

Since our WASP project concentrates on information management we first distinguish between a variety of tasks in this application domain where agents could be deployed:

- information retrieval – to collect and filter information
- information presentation – to present the information in an understandable way
- intelligent navigation – to enable the selection of relevant information

In order to realize such functionality in a Web-based context, a framework of agent technology is needed. Apart from the programming aspects, such a framework must allow for a declarative description of agent-behavior based on user preferences, including the cooperative behavior of software agents, that is how they interact with other agents.

In Section 6 we will indicate the research issues involved in developing such a framework. In particular, we will concentrate on the realization of such agents in the context of virtual environments, which (as we will argue in Section 5) imposes additional constraints and leads to additional requirements such a framework must satisfy.

3. Agents in virtual environments

Virtual environments, in particular 3D virtual worlds, are becoming a realistic architecture for providing services on the Web. 3D worlds offer more possibilities for presenting large information spaces in an attractive way than 2D presentations. As a consequence, agent-mediated services need an adequate representation in such environments, We will establish what possible function agents have in virtual environments, by briefly looking at the history of agent manifestations in such environment, and by exploring how agent technology can be deployed in virtual multi-user communities for helping visitors in information retrieval tasks and navigation. We discuss agent-like creatures in the following contexts: *autonomous characters in games* and *avatar guides in virtual worlds* in [5].

4 Autonomous characters in games

Although not widely accepted as science, the development of computer games has fueled the development of many state-of-the-art techniques. Computer controlled opponents have always been present in computer games and used to consist of only very basic logic that enabled them to mimic more complex behavior. These simple computer controlled opponents where usually not very intelligent, most of these characters simply followed a predefined path through the play-field or had some sort of simple algorithm which allowed them to chase the player.

In recent computer games, however, the development of more realistic virtual environments in which the games take place has also created a demand for more intelligent behavior by the computer controlled inhabitants of the game world. These autonomous computer controlled characters can be seen as autonomous agents intended for use in game-based virtual realities. See [5] for details.

5 A taxonomy of agents

With respect to the choice of dimensions, we must remark that even though our taxonomy has a bias towards the deployment of agents in virtual environments, the taxonomy is valid with respect to Web-based services in general. We consider the following three dimensions of web agent types.

1. **2D versus 3D**. A 2D web agent is one which is aware of http, file, and ftp protocols, whereas a 3D web agent is one which is aware of not only those protocols, but also virtual reality protocols. Typical 2D web agents provide a text-based interface, for example, information retrieval services. Typical 3D web agents are avatar-embodied guides that help visitors to navigate in virtual environments.
2. **Client versus server**. As the names imply, a client web agent is on client side, whereas server web agent is on server side. A typical client web agent can serve as a personal information assistant. A typical server web agent can serve as the front-end of web servers to offer information more intelligently.
3. **Singularity versus multiplicity**. As the names imply, a single web agent would not consider the interface with other web agents, whereas multiple web agents would interact with each other.

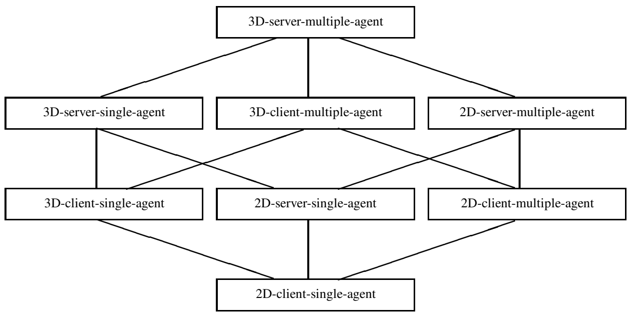

Figure 1. Lattice of Web Agents

There are different types of web agents based on those three dimensions, which consists of a complexity lattice of web agent types, which is shown in Figure 1. 3D-server-multiple-agent is on the top of the complexity hierarchy, whereas a 2D-client-single-agent is on the bottom. Those dimensions are not exhaustive. In particular, in this paper, we do not consider the dimension "stationariness versus mobility". However, the combinations based on the dimension "stationariness versus mobility" can be generalized accordingly, based on the current taxonomy. We leave it as further work. All the dimensions we consider for the taxonomy are directly web-dependent. The dimension "2D-3D" specifies the internet protocols agents have to be aware of, the dimension "client-server" is concerned with internet service modes agents have to offer, and the dimension "singularity-multiplicity" decides agent communication languages. We do not consider the functional dimensions which are not directly to the Web, like cooperating agents, problem solving agents, and negotiation agents, etc. We do not discuss the dimensions which are domain dependent, like expertise seeking agents, e-commerce agents, etc. However, our taxonomy does cover most types of those agents. They are either 2D agent or 3D agent, either client agent or server agent, either single agent or multiplity agent, or some of their combinations. Different types of web agents suggest different types of interaction modes with users and web servers. We discuss them in detail in the following.

3D-Server-Multiple-Agent is shown in Figure 2. The agents are the parts of virtual reality servers, more exactly, virtual community servers, for they interact with multiple users and agents. Users communicate with virtual reality servers via web browsers. The agents interact with each other in the virtual reality server, and communicate with web users. As agents in virtual reality, they are normally embodied by their avatars. These kinds of web agents have a great deal of application potentials, which range from avatar-embodied guides that help visitors in information retrieval tasks and navigation in virtual environments, e-commerce shopping assistant agents in 3D-virtual shops, to intelligent agents in multiple player computer games.

3D-Server-Single-Agent is shown in Figure 2 except there is only one agent on it. This kind of agents can serve as the front-end of the servers. Users do not directly communicate with servers, but with the intelligent web agent which is located on the server side. In other words, the agent may be regarded as a portal through which the server is accessed. That kind of web agents can intelligently offer or create personalized virtual environments for users, based on users' queries or requests.

3D-Client-Multiple-Agent is shown in Figure 3. The agents are located at the client side, and normally serve as users' personal information assistants. Virtual reality servers may have their own web agents. Web users communicate with servers or other users (or agents) either via

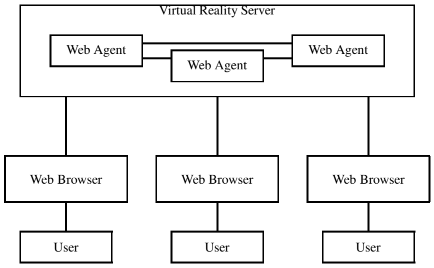

Figure 2. configuration of 3d-server-multiple-agent

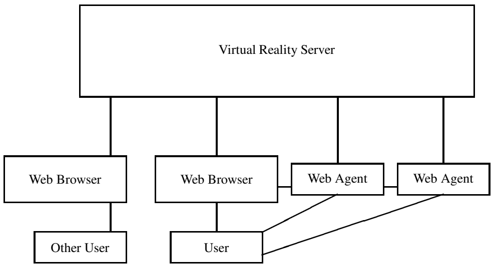

Figure 3. configuration of 3d-client-multiple-agent

web browsers or directly via web agents. Other users may also have their own web agents (which are omitted in the figure for simplicity). This type of web agents is particularly useful in the 3D chat arena, in which the agents can serve as intelligent personal assistant to help users in information gathering, and interface with other intelligent web agents.

2D-Server-Multiple-Agent is similar as its 3D counterparts, these kinds of web agents are a component of web servers. They can offer users a text-based or image-based interface for the navigation. One of the useful application for them is to serve as intelligent e-commerce sales-bots to offer more user-friendly and more intelligent services. This type of web agents can display intelligent behaviour in text-based internet games, like MUD games.

3D-Client-Single-Agent is a simplified 3d-client-multiple-agent. Its relations with other web components is like the situation shown in Figure 3, except that there is only one agent available. Similar to their multiple-agent counterparts, this type of web agents normally serves as personal information assistant in virtual environments.

2D-Server-Single-Agent is a restriction of the 3D-server-single agent, could alternatively be called *intelligent web server*. These kinds of agents normally locate at the front end of an http server to offer users more personalized messages. They can serve more efficiently and more intelligently if they are supported by powerful inference engines.

2D-Client-Multiple-Agent is analogous to the 3D-client-multiple-agent which is shown in Figure 3, apart from the more powerful user interface and display properties of the latter. One useful application of them is to act as an internet chat agent. They help users to record and analyse chat messages, to gather information and interact with other users or agents when requested.

2D-client-Single-Agent may simply be called personal (text-based)-information assistants. Its relation with servers and users is analogous to the situation shown in Figure 3, except that there is only one agent, which only has a 2D manifestation. Although they are among the simplest form of web agents, as they lack the facilities to interact with other web agents, they can still fulfill a lot of interesting and useful tasks, like gathering information on request.

6. Research issues

With the taxonomy as outlined in the previous section, we are able to indicate the research issues involved in developing a Personal Assistant for Maintaining Electronic Archives (PAMELA). We note that the PAMELA application is just one in a range of possible agent applications for the Web, and we will mention some more applications at the end of this section.

The primary requirements PAMELA must fulfill may be summarized as follows:

- autonomous and on-demand search capabilities
- (user- and system) modifiable user preferences, and
- (multimedia) presentation facilities

Evidently, with regard to our taxonomy, we have a choice of where to put the functionality, i.e. either on the client-side or server-side, whether we support a 2D (text) interface or an embodiment in 3D, and whether we realize PAMELA as a single agent or as a multi-agent system. Each of these choices has ramifications with respect to the effort of developing the application as well as the resources needed for deploying it.

Nevertheless, irrespective of a particular choice, the following aspects play a role in developing Web-aware agents, or for that matter, a framework for developing such agents:

- generic agent software components
- user interface components for managing agents
- communication primitives for agent communication

Where we position the application in our lattice of possible agent applications will, however, affect the complexity of the components mentioned above. For example, developing 3D-client-multiple agents will require advanced interface components, as well as powerful communication primitives.

To deal with the complexities involved in developing a range of agent applications, we aim at providing support on the level of the software architecture. Architectural support for agent applications must, in our view, include

- an agent-based programming language
- a high-level API for Web-aware applications
- an object-based framework for distributed agent applications
- tools for agent-based information retrieval and management

Actually, in order to offer more than the agent metaphor as a guideline for developing applications, we think it is necessary to deploy an agent-oriented programming language supporting the primitives needed for expressing both the reasoning and the domain or context in which this reasoning takes place. The ingredients for such a language, which must be distributed, object-oriented and logical are outlined in [4]. In addition to the language, an agent-framework must provide components for information retrieval, communication and presentation.

Agents in virtual environments Having an agent-oriented programming language as the core of our framework, however, does not solve all problems, especially not when we aim at developing 3D-server agent applications, which must be integrated with for example the blaxxun platform[1], which consists of a multi-user community server, as well as a powerful 3D/VRML client, that interoperates with the community server. Apart from components for information retrieval, information storage, Web access and communication, we must also specify how the particular communication structure of the community server, which uses client- and server-generated events as the primary communication mechanisms can be adapted to allow for delegating the agent-related issues to components build with our framework. Technically, this requires the development of definition of an extension API for the blaxxun Agent, and the development of an additional application server, that intercepts client-generated events. We may distinguish according to our lattice between agents that are either 2D or 3D and, independently, either located

	2D-agent	3D-agent
blaxxun agent	chat	avatar
blaxxun server	–	world
blaxxun client	–	EAI

Figure 4. Agents in the blaxxun platform

	type	blaxxun platform
lexicon search	2d-*-server	agent
gathering	2d-*-server	agent/server
presentation	3d-*-*	client/server
visualisation	3d-*-client	client/server
navigation	3d-single-server	agent/server
brokering	*-*-*	agent

Figure 5. Demonstrators of agent technology

on the server-side or client-side. In addition, we have the single/multi-agent dimension, which we ignore for the moment.

Figure 4 summarizes how an agent framework may interact with the blaxxun (client and server) platform. As concerns the blaxxun agent, we may have a 2D (single or multi) agent that acts on the server-side using the chat interface, or a 3D agent that may be represented as an avatar in the 3D world. Both types of agents require an extension of the blaxxun agent API with (preferably) intelligent facilities that aid the user in search and navigation. VRML offers an External Authoring Interface (EAI), which allows for programming a VRML world using Java. The EAI may be used to get access to agent functionality, including multi-agent cooperation, provided that such functionality is available in Java.

Demonstrators for agents in virtual environments Developing agent-technology for virtual environments is a challenging task, but probably an endeavor that is doomed to fail when one does not focus on target applications or demonstrators that deliver some useful service that can be tested in practice. To conclude our discussion of research issues, we will briefly present a list of possible target applications that may serve as the agenda for research to come.

Figure 5 presents a number of possible applications of agent-technology in virtual environments, with an indication of the type of agent involved, as well as an indication what part of the blaxxun platform is affected. An asterisks (*) is used as a wildcard to indicate where any of the alternatives is possible. As an example, for information visualisation, we will very likely need an agent that operates at the client-side, even though it uses other agents that operate at the server-side. As another example, to define a brokering service that allows visitors to come in contact with another visitor or service, any choice in the agent lattice given in Section 5 is conceivable. However, when providing navigation services, the range of alternatives is limited to one, namely 3D-single-server agents, although also in this case a multi-agent solution would be conceivable as well.

To realize any of these target applications, significant endeavor is required to model the services that are provided and for embedding these services in the blaxxun platform in a more or less seamless way. We strive for the realization of immersive virtual environments, which means that the structure and presentation of a virtual world must not be disrupted because of the availability of some service, however useful it may be. To find a seamless integration between intelligent (agent-based) services and 3D virtual worlds may, in other words, be taken as the presupposition of our research, if not a research goal in itself.

7. Conclusions

In this paper we have presented a taxonomy of agents on the Web. Particularly, the agent-lattice resulting from our taxonomy implies on ordering on conceivable agents along the dimensions 2D or 3D presentation, single or multi-agent support, and client or server-side residence. This taxonomy has been applied to indicate research issues and the ramifications target applications have with respect to a platform for realizing intelligent services in a virtual environment.

References

[1] Blaxxun, http://www.blaxxun.com, 1999.

[2] Caglayan, A., and Harrison, C., *Agent Source Book – a complete guide to Desktop, Internet and Intranet Agents* , Wiley, 1997.

[3] Cheong, F-C.,*Internet Agents: Spiders, Wanderers, Brokers and Bots*, New Riders, 1996.

[4] Eliëns, A., *Principles of Object-Oriented Software Development*, Addison-Wesley, 2000.

[5] Huang, Z., Eliëns, A., van Ballegooij, A., and de Bra, P., A taxonomy of web agents (extended version), http://www.cs.vu.nl/ ∼huang/wasp/papers/taxonomy.ps.

[6] Jennings, N. et al., A Roadmap of Agent Research and Development, *Autonomous Agents and Multi-Agent Systems I*, Kluwer, 275-306, 1998.

[7] Maes, P., Humanizing the global computer, Interview in: *IEEE Internet Computing*1(4), 1997.

[8] Negroponte, N., *Being Digital*, New Riders, 1995.