Journal of Universal Computer Science, vol. 15, no. 9 (2009), 1991-2010
submitted: 14/8/08, accepted: 25/4/09, appeared: 1/5/09 © J.UCS

Using Embodied Conversational Assistants to Interface Users with Multi-Agent Based CSCW Applications: The WebAnima Agent

Emerson Cabrera Paraiso
(Pontifical Catholic University of Paraná, Curitiba, Brazil
paraiso@ppgia.pucpr.br)

Cesar A. Tacla
(Universidade Tecnológica Federal do Paraná, Curitiba, Brazil
tacla@utfpr.edu.br)

Abstract: We have been using personal assistants (PA) coupled with multi-agent systems (MASs) in several CSCW applications. Since we are considering professional environments, where users have many tasks to perform, and where users are using several different applications at the same time (browsers, CADs, etc.), the PA interface should motivate users to keep using their assistant. To achieve this goal, we propose WebAnima. WebAnima is a web-based embodied interface agent specially designed to assist team members of a CSCW application during their daily work based on computers. In WebAnima, the intelligent behaviour is guaranteed thanks to a conversational interface and ontologies that support semantic interpretation. We believe that embodied conversational assistants will improve the quality of assistance and increase collaboration between project members. With WebAnima, we expect to acquire information that can be further processed and reused in current and subsequent projects aiming at increasing productivity. In this paper, we present the embodied conversational assistant and its insertion into an MAS designed for research and development projects. We describe the design of the agent, highlighting the role of ontologies for semantic interpretation and the dynamic behaviour of the embodied animated agent.

Keywords: Personal Assistants, Embodied Conversational Assistants, CSCW, Ontologies
Categories: I.2.4, I.2.7, H.5.2, H.1.2

1 Introduction

Several projects have been developed involving the use of a multi-agent system (MAS) for improving cooperative work based on computers ([Spinosa, 02], [Tacla, 03] and [Wu, 02]). The first reason to employ MAS is that, as in a team, an MAS consists of a group of possibly heterogeneous and autonomous agents that share a common goal and work cooperatively to achieve it. We have been using personal assistants (PAs) coupled with MAS successfully (see [Enembreck, 02], [Paraiso, 05] and [Tacla, 03] for further details). In our approach, the particular skills of a PA are devoted to understanding its master and presenting information intelligently and in a timely manner. The main goal of such an agent is to reduce the user's cognitive load. As a particular case of interface agents, PAs help users by reducing the ever-growing load of information, events and various commitments they need to handle, for instance, by learning how to organize and keep track of relevant items [Richard, 07].

The PAs can be developed so that they can be adapted to their owner, providing the necessary semantic glue to access external services (provided by service agents) uniformly [Barthès, 03]. Information can be captured on the fly, improving knowledge management. This architecture was proven in many projects ([Barthès, 03], [Enembreck, 02], [Paraiso, 06] and [Tacla, 03]) related to cooperative design supported by computers. In our approach, the PA is the only interface the system has. As a consequence, the interaction between agents (artificial or humans) is very intensive and should not waste users' time. Since we are considering professional applications, where users have many tasks to perform and where users are using several different applications at the same time (browsers, word processors, CADs, etc.), the PA interface should motivate users to keep using their assistant. Penichet and colleagues [Penichet et al., 08] state that the user interface is a key factor that influences the degree of acceptance of cooperative applications. In order to achieve this goal, we have developed a new personal assistant called WebAnima. WebAnima involves the use of conversational animated personal assistants coupled with the MAS. Even though, in its first version, WebAnima supports users with simple daily work, the main purpose of the system is to be the front end of real cooperative applications.

A conversational animated personal assistant is the result of mixing personal assistants and embodied conversational agents. Embodied conversational agents are animated anthropomorphic interface agents that are able to engage a user in real-time, multimodal dialogues, using speech, gesture, gaze, posture, intonation, and other verbal and nonverbal behaviour to emulate the experience of human face-to-face interaction [Bickmore, 04]. They are designed to develop a conversation just like a human being, as much as their intelligence allows [Sing, 06]. In WebAnima the intelligent behaviour is guaranteed thanks to a conversational interface [Kölzer, 99] and ontologies that support semantic interpretation. Each team member has a WebAnima agent that behaves according to its user's profile built on the fly. WebAnima can potentially improve the exchange of information among the participants, provide support, improve workflows and procedure controls, and provide convenient user interfaces in MAS-based CSCW applications [Paraiso, 06].

In WebAnima, we focus on agents that perform a task playing a role, for instance, as a teacher, a tutor, or as an agent who acts on behalf of the user (we ignore the types of characters that serve as desktop visual pets (e. g. cats, dogs) [Sanders, 00]). Thus, a WebAnima agent can be used in different domains, since its knowledge about the domain and tasks to perform is clearly stated in ontologies. The way an agent represents its world and the domain of its application influences how it learns, how knowledge is transferred across tasks, and how knowledge is communicated—either between agents or to a human being [Beeson, 07]. Among the several ways used to represent knowledge, ontologies are one of the most used. The key components that make up an ontology are a glossary of basic terms containing the precise specification of what those terms mean [Guarino, 98]. As a result of this approach we expect:

- to improve the quality of assistance;
- to improve collaboration between members of a project;
- to improve user's interest in using the system;
- to reduce the user's cognitive load; and

- to acquire information that can be further processed and reused in current and subsequent projects aiming at increasing productivity.

In this paper, we present the WebAnima architecture and how it can centralize and control user interaction in an MAS application. In order to contextualize, examples are based on an MAS that supports a research and development team on its daily activities. The paper begins by presenting some related work. Then, we describe the MAS architecture containing two types of agents: Personal Assistants and Service agents. After that, we present the WebAnima agent used as an implementation of PA agents. Following, we describe how ontologies are used for syntactic and semantic interpretation and for task representation when a conversation occurs between the user and the PA. We also describe the embodied animated interface. We also present some results from practical experiments we have performed. Finally, we offer a conclusion and indicate some perspectives for forthcoming work.

2 Related Work

In this work we integrated a personal assistant, a mixed-initiative ontology-based dialogue system and an embodied animated interface (concerning task execution they are mainly delegated to service agents). This is a very particular and innovative approach. Although the research on each of these three main aspects is very rich and alive, we have not find similar project in the literature. In this discussion we focus on how researchers on this field deal with information presentation and controlling.

The Smart Personal Assistant (SPA) is a research project focused on natural language interaction with personal assistant systems for use on mobile devices such as PDAs (Personal Digital Assistants) and mobile phones [Wobcke, 07]. The current SPA is a personal information management assistant that provides users with integrated access to e-mail and calendar information. Their concept of personal assistants is different from ours. In our architecture, PAs user interface with the system and services is executed by SAs. In their architecture, a user may have many personal assistants and "wrapper" agents for task providing. In SPA, there is also a coordinator agent. The coordinator is built using a BDI (Belief, Desire, Intention) agent architecture (see [Rao, 95] for more details on this architecture) in which both dialogue management and coordination of the task assistants are encoded in the agent's plans. Domain knowledge is placed in the Coordinator agent, centralizing all interpretation. That centralization is very dangerous since an interruption in the Coordinator agent may halt the whole system.

ASWAD (Agent-Supported Workflow in Public Administration) [Aswad, 02] is a European funded project that aims at providing public administrations with a unified and flexible Internet application for organizing cooperative work practices. The ASWAD tool is based on a groupware system with built in workflow management and PAs. The system includes components for calendaring, contacts management, email handling and document management. The PA enables users to filter and process information, and to automatically delegate tasks to others. The PA's interface comprises an animated avatar (a companion), accepting very simple control commands such as open, cancel, erase, etc.

Tasks have been identified as playing an important role to knowledge workers as high-level units for organizing their information [Stumpf, 07]. Stumpf et al. developed a set of intelligent tools that allow users to organize their information by labelling them with high-level units that make sense to users. They assume that the behaviour of the user at the desktop is an interleaved timeline of different tasks and each task is associated with a set of resources (e.g. documents, folders, web pages, emails, contacts, etc.) relevant to that task. Once tasks have been defined, the user can indicate to the system the name of the current task and switch to new tasks. By mining those resources, an intelligent assistant may find useful information about tasks. From the interface user's point of view, the assistant interface is very poor since each functionality is an icon in a fixed tool bar. Contrarily to Stumpf's work, we think the services provided by the intelligent agent should be controlled by an intelligent interface, capable of handling the resources as well.

3 Overview of the MAS Architecture for Knowledge Management

In order to better understand the MAS platform we use and the application context for WebAnima agents, this section describes an MAS architecture for knowledge management (KM) systems in research and development (R&D) projects [Tacla, 03]. Our main goal is to provide a system that supports collaborative work and helps to capture and to organize experience without overloading the team members with extra-work. One of the bottlenecks in KM systems is the on-line acquisition and formalization of knowledge that can be useful for later projects. WebAnima agents perform an important role in this task by gathering and interpreting user's requests producing data that once mined and formalized may reveal working methods [Tacla, 03], communities (see [Enembreck, 07][Reinhardt, 08]) and competencies. Some general requirements for the KM system that guided the project are: the system must cover as much of the R&D activity as possible; it must save time by helping the user in daily activities; it must support users in creating and sharing knowledge; it must be reliable, secure and persistent.

Initially, there are two types of agents: Service Agents that provide a particular type of service corresponding to specific skills and WebAnima agents (PAs) in charge of interfacing humans to the system. The particular skills of a WebAnima agent are devoted to understanding its master and presenting the information intelligently and in a timely manner.

In this architecture, the WebAnima agents play a major role in the KM system. First of all, they are in charge of all exchanges of information among team members. Secondly, a WebAnima agent is able to organize the documentation of its master with the help of a service agent. Finally, as R&D members have to deal with knowledge intensive tasks, they are supposed to construct their own work methods, and in this process they should remember their past experiences and, if possible, have access to other members' past experiences. Consequently, WebAnima agents must capture and represent the team members' operations, helping them in the process of preserving and creating knowledge.

The architecture contains several service agents (Figure 1). A service agent, called Project Agent, holds all the values shared by the group and, among them, the domain, documentation and project tasks ontologies. The team members can only extend the ontologies (i.e. add new subclasses). In this way, users can express their preferences, for instance, by refining a document category from the documentation ontology. So, the first thing a new user does when she/he integrates the project is to download the existing ontologies.

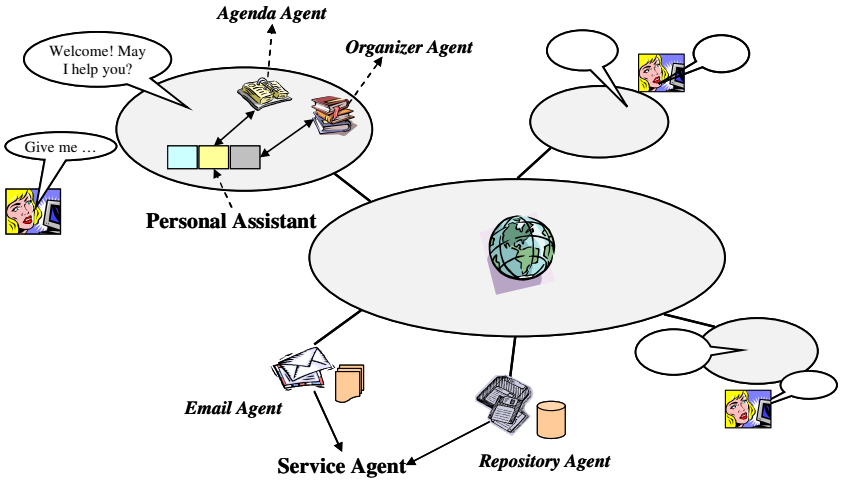


Figure 1: The MAS architecture for KM

Each WebAnima agent is supposed to help its master to organize his/her documentation. Each agent works jointly with its Organizer agent, a service agent able to build representations of the documents for later retrieval. As the architecture should be independent of any specific software tool, it integrates a service agent called Repository Agent, encapsulating the groupware that must be part of the architecture. The referred agent offers services for saving and retrieving documentation but, depending on the tool it encapsulates, it can offer other kinds of services such as WEB searches or e-mail management. There is also a set of specialized agents that perform ordinary tasks to the community of project members such as the Agenda agent and the Email Agent. The agents run on a platform called OMAS (Open Multi-Agents System) [Barthès, 03].

4 The WebAnima Agent's Architecture

WebAnima is a web-based embodied conversational assistant agent. WebAnima has, in fact, evolved from SpeechPA: an intelligent speech interface for PA in research and development projects [Paraiso, 05]. As shown in Figure 2, SpeechPA handles dialogues in natural language and was used to interface team members in an R&D project prototype. Even if SpeechPA is a mixed-initiative conversational interface, its

"static" behaviour reduces its acceptance. Due to some assumptions defined at the beginning of the project, SpeechPA follows the strategy of treating only directive speech acts [Searle, 75], reducing the number of turn-takings since some speech acts, such as acknowledgement acts (e.g. "Thank you"), are not used by the PA.

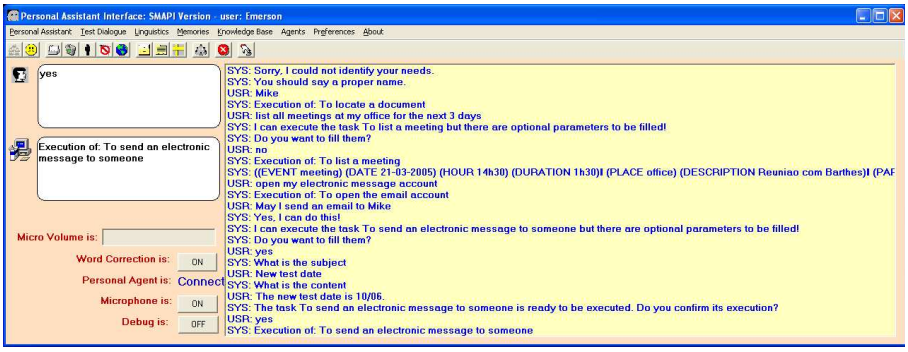


Figure 2: The SpeechPA interface

In WebAnima agent, the conversational module accepts and uses a wider set of speech acts, giving more flexibility to the agent. It also has an avatar (human-like figure) that contributes to animate the PA's use (Figure 3).

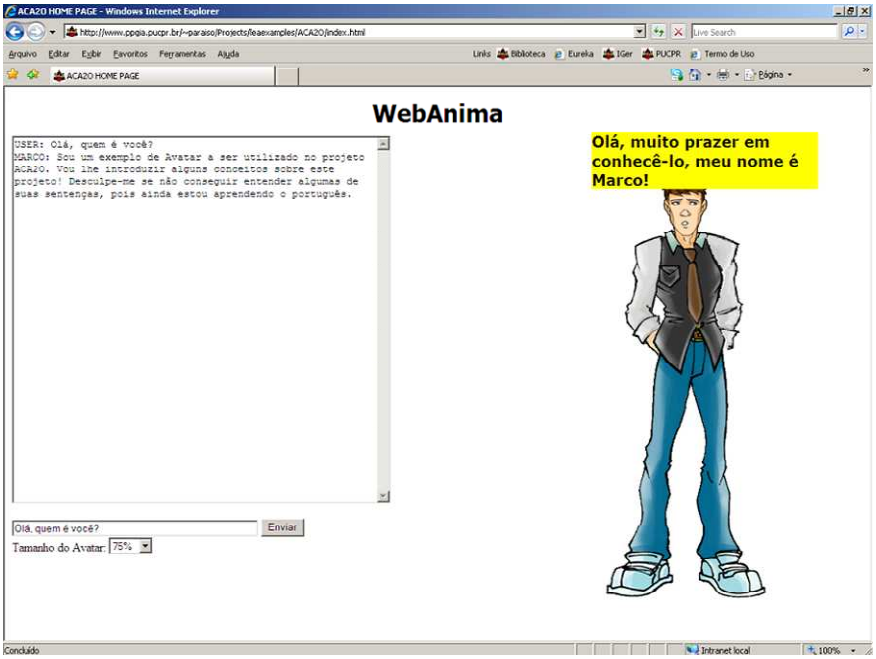


Figure 3: The WebAnima agent during a session

The WebAnima agent structure is shown in Figure 4. Each agent in the MAS has a kernel with basic functionalities. Only PAs have the user interface module since in our approach only PAs interface with users.

The role of the WebAnima agent is crucial for the effectiveness of our approach. The design and implementation of such agent is a hard task and involves many different components: dialogue controllers, natural language parsers, speech recognizers and synthesizers, knowledge manipulators, to list a few.

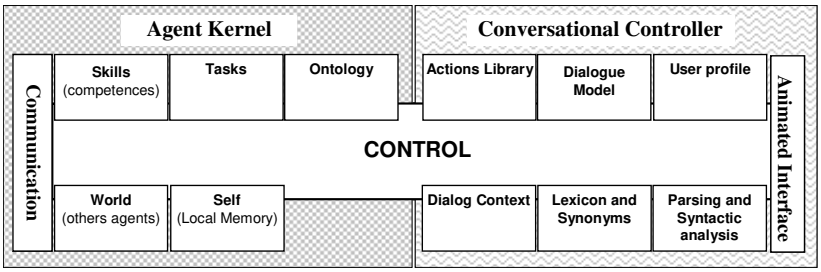


Figure 4: The WebAnima structure

Among the many types of agent models and systems that have been proposed, we have selected cognitive agents. The main advantage of cognitive agents is the possibility of designing intelligent behaviour by specifying a set of skills. In addition, in our case, such agents run independently of any particular task to solve.

Our agent is built around three main blocks: the user interface (a web-based animated interface implemented using the toolkit WebLEA [Sansonnet, 05] – see section 6 for details), the ontology-based conversational interface controller (more details in section 5), mainly responsible for controlling the dialogue (the same used in SpeechPA), and a fixed body, called the Agent Kernel. These 3 blocks are controlled by the Control module that pilots them, transferring data from one to the other.

The Agent Kernel block contains all the basic structure that allows an agent to exist. It includes a module (Communication) to establish communication with other artificial agents (SAs and PAs) using specific protocols, modules to keep the context (information) about other agents (World) and itself (Self) and modules to store the tasks in execution (Tasks). The Agent Kernel is also responsible for loading into memory the domain and task ontologies (Ontology and Skills respectively). Further information on the Agent Kernel can be found in [Ramos, 00].

Before finishing this section, it is important to highlight that the approach presented here is more suitable for a specific range of applications, in which:

- the domain is restricted and well known;
- tasks require users to have previous knowledge of the domain;
- users should be guided to accomplish complex tasks (tasks with too many sub-tasks);
- users should memorize too many steps in order to accomplish a task;
- there are some simple tasks that may be executed without the user's interference.

Each restriction presented here impacted directly upon the design of WebAnima. The application domain is defined according to an ontology, which precisely describes the domain. Tasks are also described as ontologies, enabling the PA to guide and to help users to accomplish any kind of task (the agent behaviour is task-oriented). Finally, thanks to the degree of autonomy and the presentation policy (presented in section 6), the agent may decide which task might (or might not) be executed without interference of the user.

The next two sections describe in detail the other two main WebAnima blocks: the conversational interface controller and the embodied animated user interface.

5 The Ontology-Based Conversational Interface Controller

To produce a more attractive PA, from the interface user's point of view, WebAnima incorporates a conversational interface. Conversational interfaces, as defined by Kölzer [Kölzer, 99], let users state what they want in their own terms, just as they would do speaking to another person. Conversational interfaces aim to support large-vocabulary spontaneous spoken language to be exchanged as part of a fluid dialogue between user and agent according to [Oviatt, 00] and [Komatsu, 07].

Whenever the user says something that is known as an utterance. For example, "email," "email account," or "I'd like to open my email account" are utterances.

Like most dialogue systems, we process each utterance sequentially. The process of interpreting an utterance is done in two steps: (i) parsing and syntactic analysis; and (ii) ontology application. The results are either sent to the dialogue manager continuously, or sent back to the user when they do not make sense.

The parsing algorithm replaces each utterance stem with its syntactic category (verb, noun, adverb, etc) with the help of a lexicon file and a set of grammar rules. In our application, a typical utterance could be: "I need a list of all project participants." According to our taxonomy, this is an order utterance and it can be processed by the grammar rules. If a sentence is not well formed, according to the grammatical structure, or if it is out of the domain, then it is classified as a nonsensical utterance. In this case the user is invited to reformulate his/her sentence.

The mixed-initiative and task-oriented dialogue mechanism is coordinated by the dialogue manager. It is capable of choosing a dialogue model appropriate to the beginning of a session. Each dialogue session is conducted as a task with sub-tasks. When the user requests an action, the dialogue manager tries to execute it, creating a task that is dispatched by a module called Action Looping. However, if the initial utterance lacks crucial information—e.g., an action parameter—it starts sub-tasks to complete the action list, asking for additional information from the user. The Action Looping handles GUI events and also receives calls from the dialogue manager. It is also responsible for merging all modalities (e.g. button click and speech). The Action Looping module is finally responsible for dispatching tasks to be executed (more details in [Paraiso, 06]).

In the context of an open conversation, the problem of understanding is complex, demanding a well structured knowledge basis. Domain knowledge is used here for processing the user's statements and for reasoning. To this effect, we are using a set of task and domain ontologies.

5.1 Knowledge Handling: the Ontologies

In this ontology-based conversational interface, we are using a set of task and domain ontologies, separating domain and task models for reasoning. As suggested by Allen [Allen, 01], this is interesting for domains where task reasoning is crucial. Besides, using domain knowledge separately reduces the complexity of the linguistic modules, and allows for a better understanding of statements.

Ontologies play two main roles in our PA: (i) they help an agent to interpret the context of messages sent by other agents or by the user (utterances); and (ii) they keep a computational representation of knowledge useful at inference time. The design of such ontologies must cover the user's world, in terms of entities and their relationships. In addition, the ontologies must also make the process of semantic interpretation easier, by supplying the parser with linguistics elements, such as noun synonyms, or hyponyms/hyperonyms.

5.2 Semantic Interpretation: A Case Study

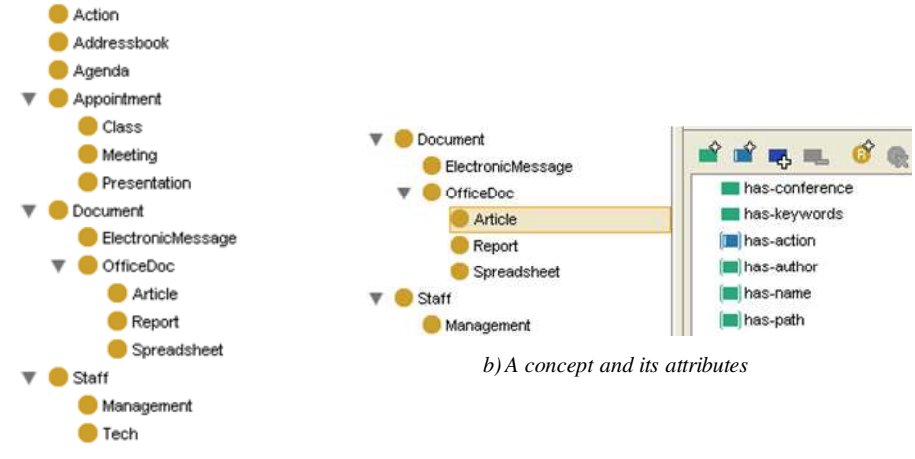
The approach to semantic interpretation presented here is based on the notion that the meaning of user's statements can be inferred by looking for concepts and their attributes. More precisely, the module responsible for applying the ontology to the statement searches for domain concepts and the list of verbs that indicate the task to be executed. The corresponding keywords are concepts of the ontology directly related to a list of actions. We believe that this approach is ideal for applications where the domain is well known and restricted.

In this paper, the ontologies are simple and short enough to understand the semantic interpretation mechanism. The concepts and their properties are organized to map the world but also to help processing natural language (by adding a list of applicable actions to each concept of the ontology). To illustrate how the mechanism works, consider the utterance:

USER: Could you list all articles about Agents?

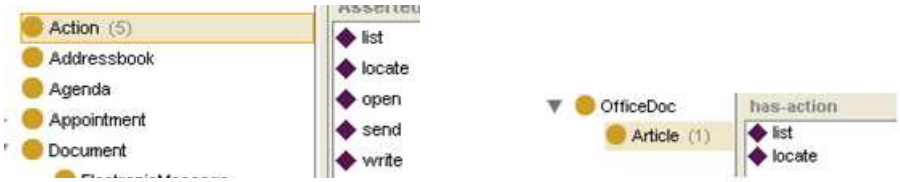
A very simple piece of ontology is shown in Figure 5a (we used Protégé [Gennari, 02] for a simplified representation), describing concepts that model a project. A project, according to the ontology, may have different types of documents, an address book, an agenda, and a list of members. A set of actions are related to each concept. Each concept may have some attributes (Figure 5b). Note that a set of actions (e.g., read, list, erase, shown in Figure 5c) may be applied to each concept, as shown in Figure 5d.

To interpret the given input, the parser checks its context. It verifies that it is a question related to the domain. To do so, it uses the domain ontology and the lexicon. Since it is a question and since it is related to the application domain, a matrix containing the list of tokens and their syntactic classification is fixed. By looking up the tokens in the ontology, one finds out that the token *list* is an action (Figure 5d). Note that it uses a list of synonyms (e.g. "list" and "enumerate") are synonyms in this context). It is also possible to find out that *Articles* is an object and *Agents* is its property. After interpreting a user's entry, we produce a formal representation of it. The formal representation is a well-formed computational formula. The process to obtain this well-formed formula is presented in [Paraiso, 08]. Next, the dialogue manager takes control of the dialogue.



a) Excerpt of the ontology

b) A concept and its attributes



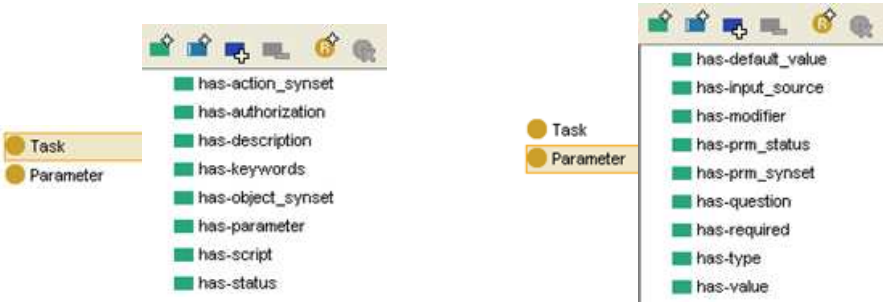
c) Action instances

d) Action list for Article

Figure 5: An excerpt of the ontology Project

Tasks in our system are represented as shown in Figure 6. A task has a set of parameters that are filled in during a dialogue session. The dialogue manager will push a task onto the stack of tasks when an utterance related to the task is given. Many tasks may be handled simultaneously (even tasks of the same type), for instance:

USER: I need to send an email to Mike Palmer.



a) Task parameters

b) Parameter fields

Figure 6: Task model

After parsing and semantic analysis, the dialogue manager is able to start a new task, since it is related to the domain (according to our first ontology presented in Figure 5). The task *To Send an Electronic Message* has some parameters to be filled in before the agent is able to execute it (each task has the structure shown in Figure 6a). One of the parameters may be the subject of the message. Since the given utterance does not contain this piece of information, the dialogue manager will request it from the user, asking him/her the question defined in the appropriate question field (as listed in Figure 6b). The dialogue manager changes the task status to pending and waits for a response from the user. When all fields are filled in, the dialogue manager sends the task for execution to a service agent.

Our platform runs in a Microsoft Windows™ environment, using the default automatic speech recognition and text to speech engines. Ontologies are OWL files. The lexicon contains every single word of the language, clustered by syntactic categories (verb, noun, adverb, etc). It was extracted from WordNet [Feldman, 98] and enriched with the list of all concepts and attributes of the domain ontology.

6 The Embodied Animated Interface

Having presented the WebAnima's ontology-based interface controller, we now focus on the third main block that composes WebAnima: its embodied animated interface. In WebAnima we used a toolkit called WebLEA. WebLEA is a 2D cartoon-like simple graphic characters technology that can be displayed and animated on web pages. WebLEA is a toolkit dedicated to the display and animation of embodied characters on web pages by using the JavaScript technology in full client mode. The 2D cartoon-like characters (Figure 7) which are used for embodying the assistant make it possible to display various postures, facial expressions and gestures [Abrilian, 02].

Figure 7: Three different WebLEA characters

In WebAnima, the character's behaviour is driven by a set of domain-independent behavioural modules which provide its capabilities for task-oriented collaboration. It is important to take into account that the agent's objective is to help users to perform their daily tasks based on computers better. This was taken into consideration when this set of behaviour modules was defined. In other words: the agent behaviour is

task-oriented, that is, the agent always tries to help its master to accomplish a task as soon as possible.

6.1 Communicative Behaviour Module

The Communicative Behaviour Module controls general communicative behaviour, such as: the use of politeness and/or humor when formulating responses, general movements (allowing the character to walk on the screen or not), the use of gestures, the size of the character, speech language (in our study case English or Portuguese), etc. These parameters are set by each user. They can be changed at any time.

As WebAnima can produce a verbal utterance directed at the user, to make it clear, the motor control automatically maintains a "speaking face" (eyebrows slightly raised and mouth moving) [Rickel, 00].

When the user is speaking, or typing an input, the agent is still quiet. In this case, it shifts the gaze to the user to indicate attention.

In this version of WebAnima, the nonverbal communicative behaviour is limited. For instance, the agent is unable to manipulate an object (e.g., pressing a button, flipping a switch). We intend to improve this module in the near future.

6.2 Context Behaviour Module

The other module is related to specific behaviour and it may have an impact on task execution or limit autonomy. This set of behaviour is defined dynamically and determined by the context. The more the agent is used, the more accurate this set is. They are initialized when a new user is created (e.g. a new engineer starts working in the project) with default values. As mentioned at the beginning of this article, each team participant has his/her own PA. Two main types of behaviour make up this set: degree of autonomy and presentation policy.

6.2.1 Degree of Autonomy

As a cognitive agent, the PA may assume some responsibilities regarding the execution of tasks. The degree of autonomy depends on the pattern defined by the user. Every time a task is selected to be executed, the PA will verify if it should demand (by its user) authorization to fire it. When writing a task (in the task ontology) the designer of the application must define if the PA should or should not request authorization before executing it (parameter "has-authorization" as shown in Figure 6a). For each task that requires authorization, the very first time it is activated, the PA will ask the user if it may assume the responsibility to execute the task without authorization the next time. Tasks such as erasing emails classified as spam or charging a new spreadsheet just uploaded in the spreadsheet database are good examples. The PA will use the same strategy to determine when to interrupt the user to present a message or to request some information needed to accomplish a task.

6.2.2 The Presentation Policy

A PA that constantly interrupts its user with boring questions or messages may drive the user to ignore it. To avoid that, a presentation policy must be defined. The presentation policy defines how information is presented and how the user is warned.

Since the PA is the only interface the user has with the system, the interaction may be stimulated by the PA or by a request made by a service agent. For instance: a service agent may inform that a printing task is finished or that an email just came in. The PA will classify each message into two categories: the ones that may be stored and displayed later and the ones that should be displayed immediately. Every message or query generated by the PA itself will be displayed immediately. Queries from service agents will be displayed as soon as possible (when the user is not busy giving information for executing a specific task). Warning messages or the confirmation of a task execution will be postponed and printed in a log window.

7 Practical Evaluation

We have performed some experiments to evaluate the embodied conversational interface and its dialogue system. Two kinds of tests were defined: a dialogue and speech related test and a usability related test.

7.1 Dialogue and Speech Related Test

In these experiments we have collected some qualitative and quantitative data shown in Table 1. Some of these measures are similar to the cost measures of the PARADISE test. PARADISE (PARAdigm for DIalogue System Evaluation), proposed by Walker et al. [Walker, 97], is a general framework for evaluating spoken dialogue agents. The framework decouples task requirements from the behaviour of an agent dialogue, supports comparisons among dialogue strategies, enables the calculation of performance over sub-dialogues and whole dialogues, specifies the relative contribution of various factors to performance, and makes it possible to compare agents performing different tasks by normalizing task complexity.

Data	**Mean (10 participants)**	**Standard Deviation**
1. Session duration (in minutes)	15.50	6.35
2. User/system turns	67.50	16.30
3. Number of accepting misleading inputs	0.1	0.31
4. Average time to execute a task (in seconds)	17.38	3.38
5. Number of words per users' input	4	0.50
6. Number of unknown words (during the whole session)	5	1.41
7. Number of offered help for actual interaction situation	1	0.81
8. Number of rejected inputs (during the whole session)	7.25	3.30

Table 1: Summary of collected data

In the experiments, a group of 10 participants interacted with the PA. Users reproduced the scenario presented in Table 2. The experiment ran over three sessions. Each participant tested the PA three times and the collected data (row 1 in Table 1) is the result of the third time. The users were given some tips on how to use the system and they were then asked to work through a series of simple training examples.

PA (9h05):	*Welcome to the Voice Personal Assistant.*
USR	Open the report on agents.
PA (9h27):	*You have a meeting scheduled for today: participant: Mike,*
USR	Send an email to Mike.
PA (9h27):	*What is the subject?*
USR	Our meeting today.
PA (9h27):	*What is the content?*
USR	We have a meeting today. Please bring your report on agent
PA (9h35):	*You have a new email: sender: Mary.*
USR	OK, could you open my email account?
PA (9h35):	*Yes.*
USR	List all meetings in my office tomorrow morning.
PA (9h41):	*You have no meetings scheduled for tomorrow morning.*
USR	What is the email address of Mary?
PA (9h41):	*The email is: mary@company.com.*
USR	Send an email to her and to Mike.
PA (9h42):	*What is the subject?*
USR	Setting a meeting for tomorrow.
PA (9h43):	*What is the content?*
USR	Dear Mike, Mrs. Mary Smith is working on personal assistant
PA (9h51):	*You have a new email: sender: Paul.*

Table 2: An excerpt of dialogue between a user and his/her PA

One can see from the results table that the domain ontology is good enough to process the inputs. Only one misleading input was accepted as an input of the domain (row 3 in Table 1). Some users had some difficulty using the system, as we can conclude by analyzing the number of times and the time elapsed to execute each task for each user (rows 2 and 4 in Table 1). One of the users accomplished all tasks in 46 turns. A turn is an exchange between speakers during a conversation: speaker A says something, then speaker B, then speaker A, and so on [Jurafsky, 08]. Another user, however, needed 80 turns to achieve the same result. This is due to the fact that some users were familiar with conversational interfaces. An important point is that all users were able to obtain the same results.

The number of rejected inputs (row 8 in Table 1) is relatively high (more than 7 inputs per user) if the whole conversation is taken into consideration. This is due to a

simple reason: testers do not have English as their native language, so their mispronunciation of an item may lead to some conversation misunderstanding.

7.2 Usability Test and Users Impressions on WebAnima

We were also interested in evaluating users' expectations and reactions concerning the system. For that, we have defined a set of general criteria as suggested by Bachmann [Bachmann, 04]: Learnability, Efficiency, Error tolerance and Accessibility.

After the third session, users were asked about their impressions concerning their experience using the system. The questions addressed general reactions of users to WebAnima. We used a 1-4 Agree-Disagree response scale (Likert Scale) for rating. The results are given in Table 3.

Evaluation Question	**Strongly Agree #**	**Agree #**	**Disagree #**	**Strongly Disagree #**
The system was easy to use	5	4	0	1
It was easy to figure out what to do	5	4	0	1
I prefer a real voice to a synthesized one	3	3	3	1
WebAnima is preferrable to a traditional user's interface	4	3	2	1
I would like to have more tasks available	7	3	0	0

Table 3: Summary of participant feedback

The users provided favorable impressions of WebAnima user interface. They were asked specially about the embodied animated character. Most of them had no experience on working (daily) in a real system with this kind of element. Thus, we cannot draw a conclusion without a more realistic evaluation, during a longer period of time.

If there were a consensus among the participants, it would be that they wanted more tasks to work on. It was not a surprise, since the scope of the dialogue was somehow limited.

7.2.1 Learnability

We began with a fundamental question: How quickly do users become productive using the system? During the experiments, we invited users to use the system prototype in three separate sessions, delayed 24 hours each one. Before the first session, users were invited to train the automatic speech recognition engine by reading a text during approximately 20 minutes. We then measured the time elapsed to start a task, which means, to select a query from the user's utterance and to query a service agent to perform the recognized task. To calculate the average time to execute a task, Tm (in seconds), we used the following equation:

$$Tm = \frac{\sum_{k=1}^{n} (D_k)}{(tt)}$$

where:

Dk is the difference, in seconds, between task identification (and respective parameters filling) and task sent to the concerned service agent;

tt is the number of executed tasks.

As expected, users became more productive after each session. The average time to start a task in the first session was 31 seconds. At the third session, time decreased to about 17.4 seconds (row 4 in Table 1). In both cases, all tasks were successfully fired.

7.2.2 Efficiency

It is essential that the user interface be easy enough to understand and use so that project members can learn to apply it effectively in their work after some introduction. Thus, another important point to study is efficiency: How easy is the system to use and to be productive? After three sessions we asked users about their impressions on using the system. The majority of the users thought the system was easy to use. Only one participant completely disagreed with the statement: "the system is easy to use". This participant added that he does not feel comfortable "talking to a machine".

As we expected, thanks to the task-oriented dialogue system, users easily figured out what to do.

7.2.3 Error Tolerance

During experiments some system errors occurred mainly related to unknown words. When an input is rejected, the system invites the user to reformulate it. During the third session, one misleading input was accepted thanks to a problem on the lexicon. This situation led the system to an unrecoverable state. We fixed the problem.

7.2.4 Accessibility

The system was designed to support the usage needs of various kinds of users, including those with special physical needs. In order to be certain of this feature, we have planned a special experiment that will take place in the near future.

A roadmap for re-testing the PA and its interface is being designed. It will evaluate mainly the improvement (or not) of knowledge capturing. This process will take several months, since some of the parameters we intend to follow (e.g. knowledge captured) are constructed from continuous manipulation and should be accumulated for an accurate estimate.

8 Conclusions

The main contribution of this work is the architecture of a web-based embodied conversational assistant to interface users with a multi-agent-based CSCW application. One of our main goals is to motivate users to keep using their assistant.

The relevance of WebAnima agents to CSCW applications - as the one presented in section 3 - is to provide data about users' activities that further process and increase knowledge sharing without overloading users with extra-work as described in [Tacla, 03]. It helps team members to articulate part of their tacit knowledge and to stimulate team members to share their knowledge. Information gathered by WebAnima agents is useful to automatically build users' profiles and to represent executed tasks and the employed resources.

Since the application is a PA, an essential feature of the user's interface was respected: predictability. It was an assumption stated at the beginning: to provide correct responses and act according to the user's command. Impossible requests, such as those out of context, are easily handled since the system uses a competence list described as an ontology.

The actual version of WebAnima does not support conversation in multiple languages. This is a special challenge in natural language based interfaces (special grammars, etc.). For the time being, we are working with English and Portuguese only.

WebAnima illustrates the enormous potential for task-oriented collaboration between team members and conversational agents in CSCW applications. According to Rickel and Johnson [Rickel, 00], although verbal exchanges may be sufficient for some tasks, we expect that, with WebAnima, many domains will benefit from an agent that can additionally use gestures, facial expressions and locomotion.

9 Future Work

One interesting topic for future work is to improve the agent's behaviour by adding a learning module to it in order to keep a more sophisticated user profile. This will allow clustering users and adapting the PA behaviour better. This is the first step to treat special multi-cultural situations.

In future work, we will also investigate the possibility of having a set of animated characters, which are applied to each component of the project according to their role in the project. In this case, the characters have to be elaborated as distinctive individuals with their own areas of expertise, personalities, visual appearance, etc. [André, 00].

Another important question is how to provide more semantically meaningful feedback to the user when he/she does not achieve the end of a task. Sometimes users do not receive a good explanation on why the task cannot be accomplished and, in the current version, this information is not available at all. This is a critical point that must be dealt with.

Finally, we are applying WebAnima in the context of small collocated software development teams. The activities of small collocated teams are often neglected by CSCW research. By analyzing preliminary requirements of small teams, it is possible to observe the need of tools to help, for instance, the elaboration of project documentation. Team members make a lot of operations during software development projects like codifying new classes, searching classes for reuse, writing reports, and taking decisions about software design. In this project, the role of WebAnima is to help to capture and represent the team members' operations (trying to gather important information for project documentation), since in a collocated group a lot of

the collaboration is informal and opportunistic [Gutwin et al., 08]. Each participant has his/her WebAnima agent, responsible for helping him/her to, proactively, look for information about the projects on development, ongoing tasks and to promote the collaboration between team members.

References

[Abrilian, 02] Abrilian, S., Buisine, S., Rendu, C. and Martin, J.-C.: "Specifying Cooperation between Modalities in Lifelike Animated Agents", Proceeding of the International Workshop on Lifelike Animated Agents: Tools, Functions, and Applications, Tokyo, Japan, (2002), 3-8.

[Allen, 01] Allen, J., Ferguson, G. and Stent, A.: "An Architecture for More Realistic Conversational Systems", Proceedings of Intelligent User Interfaces 2001, Santa Fe, NM, (2001).

[André, 00] André, E., Rist, T., Mulken, S., Klesen, M. and Baldes, S.: "The Automated Design of Believable Dialogues for Animated Presentation Teams", J. Cassell, J. Sullivan, and S. Prevost (Eds.), Embodied Conversational Agents. Boston: MIT Press, (2000), 220-255.

[Aswad, 02] Agent-Supported Workflow in Public Administration Project. (project start : 2002) Website: www.aswad-project.org/index.html.

[Bachmann, 04] Bachmann, K. L.: "Usability Requirements: Making User Satisfaction a Measure of Product Success", Proceedings of STC - Usability and Information Design, (2004), 380-381.

[Barthès, 03] Barthès, J.-P. A.: "MASH Environments for Corporate KM", Proceedings of the Knowledge Management Workshop of IJCAI 2003, Mexico, (2003).

[Beeson, 07] Beeson, P., MacMahon, M., Modayil, J., Murarka, A., Kuipers, B. and Stankiewicz, B.: "Integrating Multiple Representations of Spatial Knowledge for Mapping, Navigation, and Communication", in Interaction Challenges for Intelligent Assistants at AAAI Spring Symposium, Stanford University, CA, USA, (2007).

[Bickmore, 04] Bickmore, T. and Cassell, J.: "Social Dialogue with Embodied Conversational Agents", In J. van Kuppevelt, L. Dybkjaer, and N. Bernsen (eds.), Natural, Intelligent and Effective Interaction with Multimodal Dialogue Systems. New York: Kluwer Academic, (2004).

[Enembreck, 02] Enembreck, F. and Barthès, J.-P. A.: "Personal Assistants to improve CSCW", Proceedings of The Seventh International Conference on CSCWD, Rio de Janeiro, (2002).

[Enembreck, 07] Enembreck, F., Scalabrin, E. ; Tacla, C. A. ; Avila, B. C. "Automatic Identification of Teams based on Textual Information Retrieval", Lecture Notes in Computer Science, v. 4402, p. 1-10, (2007).

[Feldman, 98] Fellbaum, C. "WordNet: An electronic lexical database", MIT Press, Cambridge, MA, 1998.

[Gennari, 02] Gennari, J., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eiksson, H., Noy, N. F. and Tu, S. W.: "The Evolution of Protégé: an Environment for Knowledge-Based Systems Development", Report available at: http://smi.stanford.edu/pubs/SMI_Abstracts/SMI-2002-0943.html, (2002).

[Guarino, 98] Guarino, N.: "Formal Ontology in Information Systems", Proceedings of FOIS'98, Italy, IOS Press, (1998) 3-15.

[Gutwin et al., 08] Gutwin, C., Greenberg, S., Jeff Dicy, R. B. and Gregor McEwan, K. T.: "Supporting Informal Collaboration in Shared-Workplace Groupware", in Journal of Universal Computer Science, Vol. 14, No. 9, (2008), 1411-1434.

[Jurafsky, 08] Jurafsky, D. and Martin, J. H.: "Speech and Language Processing", Prentice Hall, Second Edition, 2008.

[Komatsu, 07] Komatsu, T. and Morikawa, K.: "Entrainment in the Rate of Utterances in Speech Dialogs between Users and an Auto Response System", in Journal of Universal Computer Science, Vol. 13, No 2, (2007), 186-198.

[Kölzer, 99] Kölzer, A.: "Universal Dialogue Specification for Conversational Systems", in Proceedings of IJCAI – Workshop on Knowledge and Reasoning in Practical Dialogue Systems, (1999).

[Oviatt, 00] Oviatt, S. and Adams, B.: "Designing and Evaluating Conversational Interfaces with Animated Characters", J. Cassell, J. Sullivan, and S. Prevost (Eds.), Embodied Conversational Agents, MIT Press, (2000), 319-345.

[Paraiso, 05] Paraiso, E. C. and Barthes, J.-P. A.: "An Intelligent Speech Interface for Personal Assistants in R&D Projects", Proceedings of The International Conference on CSCWD 2005 - The 9th IEEE International Conference on CSCWD, Coventry - UK, Vol. 2, (2005), 804-809.

[Paraiso, 06] Paraiso, E. C. and Barthes, J.-P. A.: "An Intelligent Speech Interface for Personal Systems in R&D Projects", in Expert Systems with Applications, Elsevier, Vol. 31, (2006), 673-683.

[Paraiso, 08] Paraiso, E. C.: "Ontology-Based Utterance Interpretation for Intelligent Conversational Interfaces", 23rd ACM Symposium on Applied Computing, Fortaleza, Brazil, (2008), 1578-1582.

[Penichet et al., 08] Penichet, V. M., Lozano, M. D., Gallud, J. A., Tesoriero, R., Rodríguez, M. L., Garrido, J. L., Noguera, M. and Hurtado, M. V.: "Extending and Supporting Featured User Interface Models for the Development of Groupware Applications", in Journal of Universal Computer Science, Vol. 14, No 19, (2008), 3053-3070.

[Ramos, 00] Ramos, M. P.: "Structuration et évolution conceptuelles d'un agent assistant personnel dans les domaines techniques". PhD Thesis (in French), presented at UTC, Compiègne, 2000.

[Rao, 95] Rao, A. S. and Georgeff, M. P.: "BDI Agents: From Theory to Practice", Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95), San Francisco, EUA, June, (1995).

[Reinhardt, 08] Reinhardt, J., Giménez-Lugo, G. A., Tacla, C. A., Albini, L. P., Sichman, J. S.: "Multiagent Systems for Design Team Assembly and Knowledge Exchange". Proceedings of the 2008 12th International Conference on Cooperative Work in Design, Xian, China, IEEE Press, v. 1. p. 349-354, (2008).

[Richard, 07] Richard, N. and Yamada, S.: "An Adaptive, Emotional, and Expressive Reminding System", in Interaction Challenges for Intelligent Assistants at AAAI Spring Symposium, Stanford University, CA, USA, (2007).

[Rickel, 00] Rickel, J., and Johnson, W.L.: "Task-Oriented Collaboration with Embodied Agents in Virtual Worlds", J. Cassell, J. Sullivan, and S. Prevost (Eds.), Embodied Conversational Agents. Boston: MIT Press, (2000), 95-122.

[Sanders, 00] Sanders, G. A. and Scholtz, J.: "Measuring and Evaluating of Embodied Conversational Agents", J. Cassell, J. Sullivan, and S. Prevost (Eds.), Embodied Conversational Agents, MIT Press, (2000), 346-373.

[Sansonnet, 05] Sansonnet, J-P., Martin, J-C and Leguern, K.: "A Software Engineering Approach Combining Rational and Conversational Agents for the Design of Assistance Applications", T. Panayiotopoulos et al. (Eds.): IVA 2005, LNAI 3661, (2005), 111 - 119.

[Searle, 75] Searle, J. R.: "A Taxonomy of Illocutionary Acts", in Proceedings of Language, Mind and Knowledge, Vol. 7, University of Minnesota Press, (1975), 344-369.

[Sing, 06] Sing, G., Wong, K., Fung, C. and Depickere, A.: "Towards a More Natural and Intelligent Interface with Embodied Conversation Agent", Proceedings of International Conference on Game Research and Development, (2006), 177-183.

[Stumpf, 07] Stumpf, S., Burnett, M.M., and Dietterich, T.: "Improving Intelligent Assistants for Desktop Activities", Proceedings of the AAAI 2007 Spring Symposium Interaction Challenges for Intelligent Assistants, Stanford University, CA, USA, (2007), 119-121.

[Spinosa, 02] Spinosa, L. M., Quandt, C. O. and Ramos, M. P.: "Toward a Knowledge-based Framework to Foster Innovation in Networked Organisations", Proceedings of The Seventh International Conference on CSCWD, Rio de Janeiro, (2002).

[Tacla, 03] Tacla, C. A., Barthès, J.-P.: "A Multi-Agent System for Acquiring and Sharing Lessons Learned". Computers in Industry, v. 52, (2003), 5-16.

[Walker, 97] Walker, M.., Litman, D. J., Kamm, C. A. et al.: "PARADISE: A Framework for Evaluating Spoken Dialogue Agents", Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, (1997).

[Wobcke, 07] Wobcke, W. R., Nguyen, A., Ho, V. and Krzywicki, A.: "The Smart Personal Assistant: An Overview", Proceedings of the AAAI 2007 Spring Symposium Interaction Challenges for Intelligent Assistants, Stanford University, CA, USA, (2007), 135-136.

[Wu, 02] Wu, S., Ghenniwa, H. and Shen, W.: "User Model of a Personal Assistant in Collaborative Design Environments", Agents in Design. MIT, Cambridge, (2002), 39-54.